INFOCUS

Windows Forensics: A Case Study, Part One

by Stephen Barish
last updated December 31, 2002

It's a security person's worst nightmare. You've just inherited a large, diverse enterprise with relatively few security controls when something happens. We all try to detect malicious activity at the perimeter of the network by monitoring our intrusion detection systems, and watching attackers bang futilely on our firewall. Even those attackers tricky enough to slip through the firewall bounce harmlessly off our highly secured servers, and trip alarms off throughout the network as they attempt to compromise it. Reality is usually somewhat different: most of us simply don't have the tools, or at least we don't have expensive, dedicated tools. But we do have ways to stop the pain.

This article is the first in a two-part series that will offer a case study of forensics in a Windows environment. This installment will offer a brief overview of the detection and analysis of attack an attack incident. The second installment will look at continue to look at network traffic analysis techniques and will resolve a hypothetical attack scenario.

Although 2002 has been a relatively quiet year for network compromises, there have been quite a few new attacks released, and a fairly significant number of incidents as a result. For the purposes of this discussion, I've blended a number of these incidents together to create a hypothetical Web-hosting company, Web4Sale.com, to demonstrate some of the techniques I've used this year in combating intrusions.

Detection of Potential Incidents

One of the most significant indications that you have a problem in your enterprise is unexpected traffic volume in unusual places. Consider our Web-hosting company as an example. The company's revenue is tied to how many Web sites it can host. This usually translates into *huge* public address space, and large numbers of systems. It's reasonable to expect that this portion of the enterprise will always have high traffic loads on it. But network management and monitoring rarely travels these connections, as even the helpdesk is a little nervous about broadcasting the root password across the Internet nowadays. What you will usually see are dual-homed servers, with a public interface to serve clients, and a private interface allowing management and control of the server from a trusted source inside the enterprise. If you start seeing large volumes of traffic here, you have a major problem: the attackers are already deep inside the network. This is actually one of the more common ways to detect a "live" attack in progress.

Another major indication of trouble, and a way to detect compromises without robust IDS or firewall solutions, is to look for unexpected types of traffic. SANS calls this, "playing the home field advantage," and it's based on the simple idea that we know our networks better than our adversaries do. If we see something abnormal, we ought to be concerned and investigate. This is really the second step when you see large or unusual traffic patterns: determine what kind of traffic it is, and see if it is "normal". If you see a lot of NetBIOS on a management interface that normally communicates via SSH or SecureCRT, you may have a problem.

Of course, in order to determine whether any activity or traffic is anomalous, we must understand our networks well. There are some great tools that are freely available on the Internet to help you characterize and understand your network. In most cases, all you need is a laptop running some flavor of Linux, and you can get the rest for free. While this won't be an integrated enterprise solution, you can build a quick and dirty toolkit that will help you solve problems when they occur. Some of the more

useful tools in my toolkit are:

- Ethereal - Ethereal will allow you to capture and graphically inspect network traffic, and can be run under Windows or Linux.
- EtherApe EtherApe builds a "talkers map" for a given network segment, and is a great tool to help characterize "normal" traffic.
- tcpreplay - tcpreplay lets you replay captured traffic and control the speed at which it flows through another program.

I frequently use tcpreplay in combination with EtherApe to watch traffic maps develop and look for anomalies. Again, these tools and others like them are freely available, and don't really take long to learn to use.

One drawback to these tools is that they are time-intensive from an analysis perspective and it takes time to learn to use all of their strengths. However, the advantage is that we can use assets that already exist to help us understand the network's normal behavior. Then, when we see major deviations from "normal" we can infer that we might have a problem. And the technique works even if you aren't a Web-hosting company or an application service provider, since virtually every network has at least one person responsible for making sure it works. Consider that poor helpdesk person or the engineer in the Network Operations Center (NOC) to be one of your IDS sensors, and you'll have a leg up on the opposition.

Identifying the Attack

Every Internet-connected network will come under attack eventually and, unless your enterprise is extremely unusual, one of those attacks will eventually succeed. The key to minimizing the damage is in understanding what happened, how the attacker compromised your system, how major the scope of the compromise was, what recovery options are available to eliminate the attacker, and the vulnerability from the enterprise.

Consider our Web-hosting example again. Thanks in part to good preparation and relationship building with the helpdesk and NOC staff, and partly to good luck, we saw a major traffic spike on the management interfaces of several servers in the Web4Sale.com production environment. Web4Sale.com, like most large providers, manages and maintains their network from a central point in the enterprise using a private network (10.20.0.0/24). This Class C equivalent address block is the only group of addresses we expect to see talking on the management interfaces of the production Web servers. Armed with this knowledge, we began trying to find out what was happening in the enterprise.

The first step in identifying what was going on was to look at the suspicious traffic. Under normal circumstances, we expect to see traffic on the management interface to and from 10.20.0.0/24. We also expect this traffic to be encrypted via SSH, so anything that doesn't match that profile is at least suspicious. Using a Linux laptop and tcpdump, we selected a host to monitor and start sniffing packets. To accomplish this in the switched environment, we plugged into a mirror port on a Cisco 2500 series switch and mirrored the traffic we were interested in:

```
tcpdump -i eth0 -s 1500 host winbox.private.com
```

where **eth0** was the capture interface and **winbox.private.com** is the address of the monitored server's management interface.

The data we got back was extremely interesting. Immediately we notice two anomalies. First, there was a tremendous volume of NetBIOS traffic on the private interface:

```
10/02/02      08:27:18      netbios.public_ip.com      137    ->     winbox.private.net    13
10/02/02      08:27:19      netbios.public_ip.com      137    ->     winbox.private.net    13
10/02/02      08:27:20      netbios.public_ip.com      137    ->     winbox.private.net    13
```

Second, there is a public IP address that is showing up in the traffic (**netbios.public_ip.com**).

Our first rule is to use the home field advantage. Our knowledge of the network indicated that NetBIOS traffic has no business being on the management interface. Worse yet, this is a private network that is not supposed to be routable from the Internet. So why is **netbios.suspicious.com** sending traffic to the private interface of **winbox.private.com**? At this point, it was pretty clear that some type of unauthorized use was occurring. The next step was to identify what was being done on **winbox.private.com** and determining whether or not it was compromised from outside the enterprise, or whether someone on the inside was using it inappropriately. It was time for inspection of the host itself.

Host-Based Forensics

The most important thing to understand when you start examining a system forensically is that nothing about the system under examination can be trusted. We treat every host we examine as if it was fully compromised, root-kitted, and aggressively monitored by the attackers in case of examination. One of the first things we need to decide is whether or not the system needs to come down. Unfortunately there is no hard and fast rule here - it's a judgment call. The investigator has to weigh the possibility that the system is being used to cause additional compromise against the risk of losing volatile data in memory. In our case, since an attacker was actively using the system under examination, we made the decision to leave the system up and running while we collected information about our adversary's methods and intent.

Forensics is really the art of obtaining, recording, and tracking information about a possibly criminal activity for the potential use in court at a later date. This means we have to take every precaution to make sure the data we collect is accurate, trusted, and is not modified from the time of collection onward. By recording each step in the collection and processing of forensic data, and tracking its movement, who accessed it, and what was done to it, we help preserve the "Chain of Custody". This is a non-trivial task that is very well discussed in any number of books on computer forensics, and your general counsel can help you if you have detailed legal questions. In general, the best practice is to collect evidence using trusted tools, save data to removable media, and ensure the data can be authenticated.

The first challenge is the tools. In reality, almost everything we would need to examine **winbox.private.com** is available in the Windows 2000 operating system or the Windows 2000 Resource Kit. However, as we cannot trust a potentially compromised device, we need to build our own trusted version of the tools we need. Simply obtaining a clean copy of the operating system (preferably one that has never been connected to the compromised network) and burning copies of certain files to a CD-ROM will accomplish this goal. Some of the tools you'll need will include:

- at.exe
- cmd.exe
- dir.exe
- ipconfig.exe
- nbtstat.exe
- net.exe
- netstat.exe
- nslookup.exe
- route.exe
- tracert.exe

There are also some freely available tools in the Internet you will want to use. One of the most important is **md5sum.exe**, a port of the Linux command of the same name that can be used to determine whether or not the evidence has been modified since it was collected. The best practice is to run a given tool, capture the output on a floppy, then immediately generate an MD5 sum of the data, with the objective of having the file creation times be nearly identical. The important thing to remember is that until the MD5 sum is generated, we have no guarantee that the data was not manipulated. If the attackers are ever prosecuted, a time-gap between data collection time and the MD5 sum time can be extremely damaging to the prosecution's case.

Other tools of use will include Fport.exe from Foundstone and a variety of tools including the following:

- pslist.exe - a tool to list processes from the Windows 2000 command line.
- psservice.exe - a tool that associates services with process Ids for Windows 2000.
- psfile.exe - a Windows 2000 tool similar to lsof does under Linux.
- psloggedon.exe - a tool that associates users with running processes.
- listdlls.exe - a tool to list which DLL files are being used by running processes

A simple Internet search will yield these tools and many more of use during forensic examination of Windows 2000 hosts.

In the case of this investigation, the first step was to identify who was logged onto the system, what resources were being shared, and what processes were running. All commands were run from the CD-ROM (E:\) and data was saved to the floppy (A:\):

```
E:\nbtstat -a winbox.private.com > a:\nbtstat-a_output.txt
E:\md5sum a:\nbtstat-a_output.txt > a:\nbtstat-a_output.md5
```

This pattern was repeated for each command below, and the output on the floppy was later analyzed to determine the following information. There are many good texts on computer forensics including The Anti-Hacker Toolkit by Keith Jones and Incident Response by Kevin Mandia (among numerous others), as well as excellent articles on SecurityFocus about specific techniques to employ in either a Windows or Unix environment. However, this article is less about tools, and more about techniques. what we need to look for while we execute these tools.

What to Look For?

The real issue in forensic analysis, as with any response-oriented activity, is to determine what happened, who the culprit was, and what the impact of the event has been. The tools discussed above are extremely useful in this analysis, but ultimately it is the skill of the investigator that determines whether or not a case will be resolved. While most experts agree the that field of computer forensics is at least as much art as science, there are a handful of things everyone agrees are important things to look for:

Objective	Tool / Technique
Identify unusual processes	pslist, psinfo, psfile
Identify unusual listening ports	netstat, Fport, psservice
Identify unusual open files	psfile, listdlls, Fport
Identify logged in users	psloggedon, nbtstat
Identify process owners	psloggedon
Examine routing tables	netstat, route
Examine temporary files	dir, type
Identify suspicious directories/folders	dir, Explorer

We are attempting to reconstruct the scene of the crime, so to speak. Remember, one of our key advantages over the adversary is that we have the home field advantage - we know our networks and systems better than those attacking us. Armed with the knowledge of what constitutes "normal", we can make a cursory examination of a compromised server in less than a half-hour. The off-line analysis often takes much longer, particularly when you are examining directory structures looking for hidden directories, or items that shouldn't be there. In addition, we always need to remember to look for clues in the registry that might identify what our attackers were trying to accomplish. Temporary files can be examined with a sector editor for code fragments or clues to other activity. Batch files (*.bat) should be examined to make sure they haven't been modified or created simply to further the attacker's goals. And finally, in most Windows environments there is at least minimal logging going on using the Windows Event Logs and Security Logs.

In the Web4sale case, **winbox.private.com** was examined using trusted versions of netstat, route, nbtstat, hostname, net, and dir. Additionally, we used Fport, pslist, psloggedon, and psservice to identify the owners of a variety of suspicious processes that were detected. The excerpts below show some of the data collected during this investigation.

```
E:\hostname
Winbox.private.com


E:\nbtstat -a winbox.private.com
NetBIOS Remote Machine Name Table

Name   Type   Status
---------------------------------------------
WINBOX <00>   UNIQUE Registered
WINBOX <02>   UNIQUE Registered
PROD   <00>   GROUP  Registered
PROD   <1E>   GROUP  Registered
..__MSBROUWS__        <01>   GROUP  Registered
ADMINISTRATOR <03>   UNIQUE Registered

MAC ADDRESS = XX-XX-XX-XX-XX-XX


E:\net session

Computer      User name     Client Type   Opens  Idle time

-------------------------------------------------------------
\\TGT1 ADMINISTRATOR        0      00:00:27
\\TGT2 ADMINISTRATOR        0      00:00:15
\\TGT3 ADMINISTRATOR        0      00:00:23
\\TGT4 ADMINISTRATOR        0      00:00:05


E:\Fport.exe
Fport v2.0 - TCP/IP Process to Port Mapper
Copyright 200 by Foundstone, Inc
http://www.foundstone.com

Pid    Process              Port   Proto  Path
420    svchost       ->     135    TCP    C:\WINNT\system32\svchost.exe
8      System ->     445    TCP
888    MSTask ->     1025   TCP    C:\WINNT\system32\MSTask.exe
8      System ->     1027   TCP
8      System ->     445    UDP
430    svchost       ->     80     TCP    C:\Program Files\Apache\httpd.exe
1625   servu  ->     3215   TCP    C:\Client_Data\Inetpub\_vti-bin\ \servu.exe
```

While I have not shown all the output of each tool, there are some important clues in this data. First, we have confirmed the hostname of the target (winbox.private.com) and have gathered some information about who is logged in, what processes are running and which services are listening on the device. Given this output, there are several things of concern to us from an investigative perspective. The first thing that leaps out is the output of the `net session` command, which clearly shows NetBIOS sharing between **winbox.target.com** and a number of other servers using the private address of each server. We know this is abnormal, as the private network is intended for management only, and resource sharing using MS Windows File and Printer Sharing is specifically unauthorized by policy. We also see that in each case, the illicit connections were made using the authentication credentials of the local Administrator on the target machine - this is a very bad thing. Worse, the output of **Fport** indicates **winbox.private.com** is running what appears to be an FTP server, on an ephemeral port (greater than 1024), whose source is in an extremely unusual location (**C:\Client_Data\Inetpub_vti-bin\ **), which appears to be a hidden directory. This is particularly suspicious since **winbox.private.com** is running Apache as the Web server, and the directory structure for the FTP server seems to be hidden in a directory that is normally associated with Microsoft IIS.

By recursively listing the hidden directory (**E:\dir /s /a C:\Client_Data\Inetpub_vti-bin\" "\ /p**)

we were able to find the executable version of the FTP server, its configuration file, and a number of interesting batch files, configuration files, and tools. Of particular interest was the batch file titled "VFS_MNT.BAT" (partial listing below).

```
net use F \\tgt1\c$\WINNT\system32\ \_vti-bin\ /user:Administrator AdminPass
net use G \\tgt2\c$\WINNT\system32\ \_vti-bin\ /user:Administrator AdminPass
net use H \\tgt3\c$\WINNT\system32\ \_vti-bin\ /user:Administrator AdminPass
net use I \\tgt4\c$\WINNT\system32\ \_vti-bin\ /user:Adminstrator AdminPass
```

There is an alarming trend here. First, the attacker has mounted a hidden directory on each of the targets with local administrator privileges, allowing the attacker to create a virtual file system for the illicit FTP server. Of course, in reality sharing the Administrator password across systems this way is extremely poor practice, but in this case it was the norm. Worse, the attacker now has the ability to serve data hidden across multiple systems via one or potentially more illicit FTP servers hidden in the enterprise.

Putting it All Together

This case study is really a blending of a number of incidents examined in the latter part of 2002, and doesn't really represent any single incident. However, all of the incidents observed shared certain common traits: 1) large enterprises supported by extremely high bandwidth Internet connections; 2) largely Windows NT and Windows 2000 enterprises; 3) persistent compromise of Administrator and Domain Administrator accounts; 4) widespread use of a distributed 2-tier FTP server, where the FTP root directory structure was comprised of a virtual file system of shared drives. Our mythical Web hosting company, Web4Sale.com, suffers from this same set of common criteria, and like most of the real-world incidents, the attackers are serving both *warez* and porn.

At this stage of the investigation, we've discovered the compromise (although we have yet to identify the compromise method), identified the post-attack "fingerprint" of this particular group, and have begun to understand what is happening in the enterprise. Should we need to initiate corrective action at this time, there are a number of things we can do to end the current compromise, starting with changing the Administrator passwords, and restricting NetBIOS using packet filters on the switches supporting the Web4Sale.com enterprise. However, before we start with the eradication phase of our incident response, we really need to complete the identification phase: we have yet to identify the initial compromise method, or to identify the scope of the compromise! In the next installment of this series, we'll look at network traffic analysis techniques to continue our response, and resolve these issues.