Global Information Assurance Certification Paper

ADVANCED INCIDENT HANDLING AND HACKER EXPLOITS

OPTION 2 – DOCUMENT AN EXPLOIT, VULNERABILITY OR MALICIOUS CODE

SQLEXEC

Chuck Crawford

May 28, 2001

INTRODUCTION

This paper will explain the SQLEXEC v 1.0 exploit. The SQLEXEC was written in Visual C++ 6.0 by Egemen Tas. Although this program was written for administrative uses, it can easily be used as a hacker exploit.

Databases are a big piece of corporate networks. Internet servers, browsers, and remote workstations can access them. This exploit can allow an unauthorized user to run command line queries, command line statements, batch files, and scripts. All of these can cause serious damage to a company's SQL server, or even worse gather private information such as credit card numbers or passwords.

The default installation of Microsoft SQL Server version 7.0 does not assign a password to the "sa" login. This, for obvious reasons raises several red flags. If the SQLEXEC program is run in "Default" mode, it will try to exploit this vulnerablitiy. Even if the system administrator has assigned a password to the "sa" login, this program can still be run thru other security flaws in Microsoft SQL, IIS, and MDAC.

Unless a company has opened a port or created a proxy on their firewall, I would see this exploit as more of a threat from an internal hack. Thru social engineering, a hacker could easily find out the name of a SQL server that his or her company uses. Once that is accomplished, the user can simply try and run the exploit. As I will mention later, there are other ways of retrieving the "sa" password of a SQL server as well.

EXPLOIT DETAILS

NAME: SQLEXEC v1.0

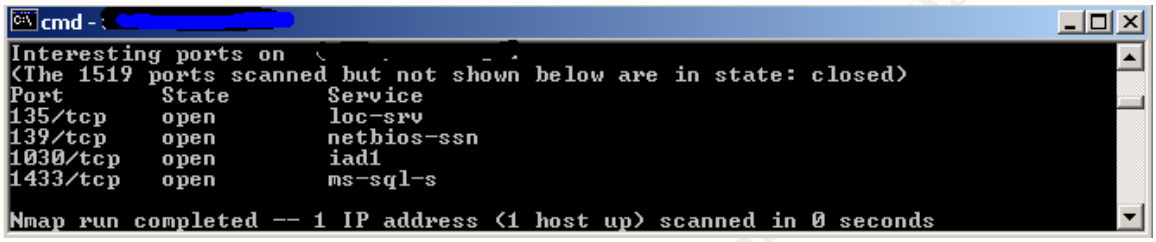
OPERATING SYSTEMS: Windows NT Server 4.0

PROTOCOLS: TCP port 1433

VARIENTS: SQLEXEC for Oracle

BRIEF DESCRIPTION: This utility uses exploits in the default installation settings of Microsoft SQL 7 to gain access to a SQL 7.0 server thru a command shell interface. This utility can be run against the standard edition as well as the Enterprise edition.

The default use of SQLEXEC uses TCP on port 1433 to connect to a Microsoft SQL server remotely or locally. This is the default port SQL server uses for communication.



As with most client server applications, we can actually follow the OSI model on a typical connection.

- *APPLICATION*: The SQL server provides the network services for end user applications. In this case SQLEXEC.
- *PRESENTATION*: Formats the code and can make sure that the data that arrives from the network can be used by the application. Extremely useful in situations where different platforms are used.
- *SESSION*: This layer establishes and maintains the session between the applications. Password authentication, etc happen in this layer.
- *TRANSPORT*: This is where the TCP protocol resides. We will discuss this layer further later.
- *NETWORK*: This layer is where routers reside. One of its purposes is to determine the most efficient path for data to travel. IP resides at this layer.
- *Data Link*: This is where the MAC address on the NIC card is used. It prepares the packet for transmission.
- *Physical*: This is where the cabling resides. Provides the electrical and physical means for the transportation.

As you can see from the OSI model, TCP or Transfer Control Protocol runs in the transport layer. This layer segments and reassembles data into streams. TCP provides very reliable point-to-point communication, and Full Duplex Communication.

The way TCP establishes and terminates communication is thru a "three-way handshake". This ensures clear agreement despite packet loss, duplication and delay.

THREE-WAY HANDSHAKE
Connection

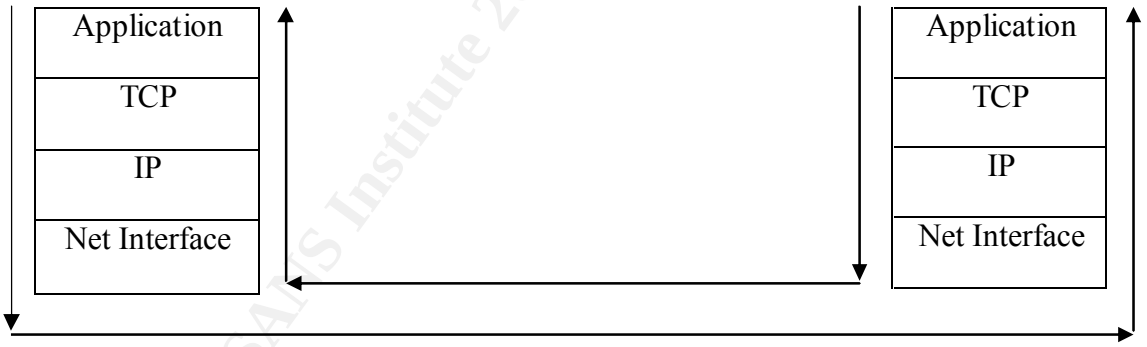
- TCP Issues a SYN
- Responding host sends back an SYN/ACK
- Finally an ACK is issued to complete the connection

THREE-WAY HANDSHAKE
Terminate Connection

- TCP Issues a FIN
- Responding Host sends back a FIN/ACK
- Finally an ACK is issued to end the connection

Once TCP begins the handshake, the data is transferred in the Network layer using IP datagrams. When the responding host recieves the IP packets it moves it up to the Transport layer for the SYN/ACK and then sends that data back down to the Network Layer over to the requesting host. The process is then reveresed again for the ACK statement. As you can see from the chart, this is the same process for connection termination.

TCP establishes a "virtual connection" from one application to another application on a remote computer. This is only virtual, because it is a software-oriented connection. It uses IP datagrams to carry the data to the remote computer. From looking at the diagram, basically the IP datagrams are moved along the Netwok Interface to the remote workstation where the connection begins.



If you look at a TCP data thru a sniffer you will notice that the header on a typical IP datagram is 20 bytes. TCP has no idea what the contents of the bytes are. Each TCP segement contains the source and destination port number to help identify the application that is doing the communicaitng. The contents of the bytes are left up to the application to to read and interpret.

Here is an excerpt thru the "eyes of a sniffer" from a typical Telnet Connection. Telnet normally runs on port 23, for this example I will use 1024.

15:27:41:883816 < 172.16.2.76.1807 > 172.16.2.77.1024:S 159942:159942(0) win 65535 <mss 1460 (DF) [tos 0xc4]

15:27:41:883816 < 172.16.2.77.1024> 172.16.2.76.1807:S 1890001877:1890001877 (0) ack 159943 win 5 840 <mss 1460 (DF)

15:27:41:883816 < 172.16.2.76.1807> 172.16.2.77.1024 . (0) ack 1 win 65535 (DF) [tos 0Xc4]

15:28:15:363816 < 172.16.2.76.1807> 172.16.2.77.1024:F 1:1(0) ack 1 win 65535 (DF) [tos 0Xc4]

15:28:15:363816 < 172.16.2.77.1024> 172.16.2.76.1807:F 1:1(0) ack 2 win 5840 (DF)

15:28:15:383816 < 172.16.2.76.1807> 172.16.2.77.1024: . 2:2 (0) ack 2 win 65535 (DF) [tos 0Xc4]

The first piece of the sniffer output shows the actual connection request. After the time, it lists the source IP address along with it's port number. Then the destination IP address is listed along with its' port number. The "S" shows the SYN request. This as stated above is the beginning of the three-way handshake. Next on the portion is the starting sequence number and the ending sequence number, followed by the window size of the packet. After this, the maximum segment size is listed along with the type of service.

The next piece of the sniffer output again shows the source IP address, which this time is the server being telnet'ed to, and its port number. Then it shows the destination IP address and port number. Next is the Syn flag. Following this are the starting and ending sequence numbers. Then the Ack flag is listed with a sequence number that is 1 higher than the sequence number issued by the original Syn request. Finally you have your window size packet and your maximum segment size listed. This packet illustrates the second part of the three-way handshake, the Syn-Ack.

Finally the third piece illustratres the Ack portion of the handshake. Again it shows the time, source and destination IP address's along with the port numbers that the communication is going thru. Following that information is the Window size parameter, and the Type of service.

The last three lines shown, illustrate and typical connection termination. Here you will notice the F instead of the S. These are the Fin statements. Once again these look pretty similar to the Connection sequence of the handshake.

There are many other areas of the TCP protocol that can be examined. For instance the time of connection establishment, maximum segment sizes, TCP Half-close, or full-close are just a few examples of the different portions of this protocol.

Keypoints to remember about TCP is that it is a very reliable full duplex protocol. A connection is established before any data begins to transfer.

SQLEXEC VARIANTS

SQLExec was written in Visual Basic. I am sure there are plenty of other variants out there in the hacker and user community. However, there is one variant that I know of that appears to work on an ORACLE database server. Links to this version can be found at

http://processweb.cs.man.ac.uk/doc/SQLexec.html

http://sweetpea.cs.man.ac.uk/doc/SQLexec.html

For the purpose of this paper we will focus on the SQLEXEC program designed for Microsoft SQL servers. The similarities between the two different scripts are amazing.

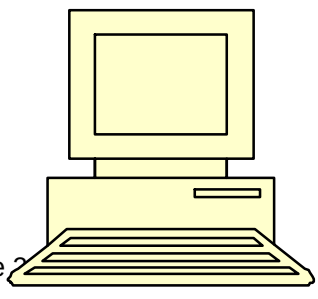
HOW THE EXPLOIT WORKS

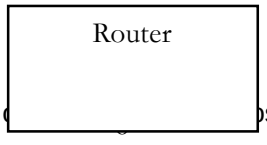
The SQLEXEC exploit can be easily run. You can run this remotely, on the LAN or from the server console itself. Basically anywhere you have access to the SQL servers network you can run this exploit.

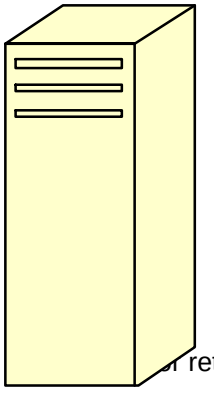
Once the exploit is ran and the user is connected to the SQL server, they will have access to the xp_cmdshell and be connected to the Master Database of the SQL server.

Basically, the SQLExec program takes advantage of several default installation parameters with SQL Server 7.0. First there is no password assigned to the "sa" login ID. Even if the System Administrator assigned a password , in many cases it is not that difficult to find it out. Second a simple port scan from a utility like Nmap will show what port SQL is listening on. 1433 is the most common and default port. Third, the default database for the "sa" login ID is the master database. This is the most important database on a SQL server.

DIAGRAM







HACKER SQL SERVER

The hack can take place over the Internet, on a LAN, or on the console itself. This is a very simple exploit on Microsoft SQL's default installation. Since most companies won't have an open port on their firewall for SQL connections, this exploit is probably a greater threat for an internal hack, unless the intruder has somehow gained access to the companies internal network.

To run the program, the hacker issues or types the SQLEXEC statement plus the SQL server name at the command line of their workstation and it will connect that user to the SQL server. He or she does not need to know the IP address of the SQL server.

Here is an excerpt from TCPDUMP after I issued the SQLEXEC command to a SQL Server:

10:55:18.305637 USERMACHINE.1952 > SQLSERVER.1433: S 2486413029:2486413029(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)

10:55:18.306088 SQLSERVER.1433 > USERMACHINE.1952: S 1926380019:1926380019(0) ack 2486413030 win 8760 <mss 1460> (DF)

10:55:18.306146 USERMACHINE.1952 > SQLSERVER.1433: . ack 1 win 17520 (DF)

10:55:18.312093 USERMACHINE.1952 > SQLSERVER.1433: P 1:198(197) ack 1 win 17520 (DF)

10:55:18.314845 SQLSERVER.1433 > USERMACHINE.1952: P 1:405(404) ack 198 win 8563 (DF)

10:55:18.453918 USERMACHINE.1952 > SQLSERVER.1433: . ack 405 win 17116 (DF)

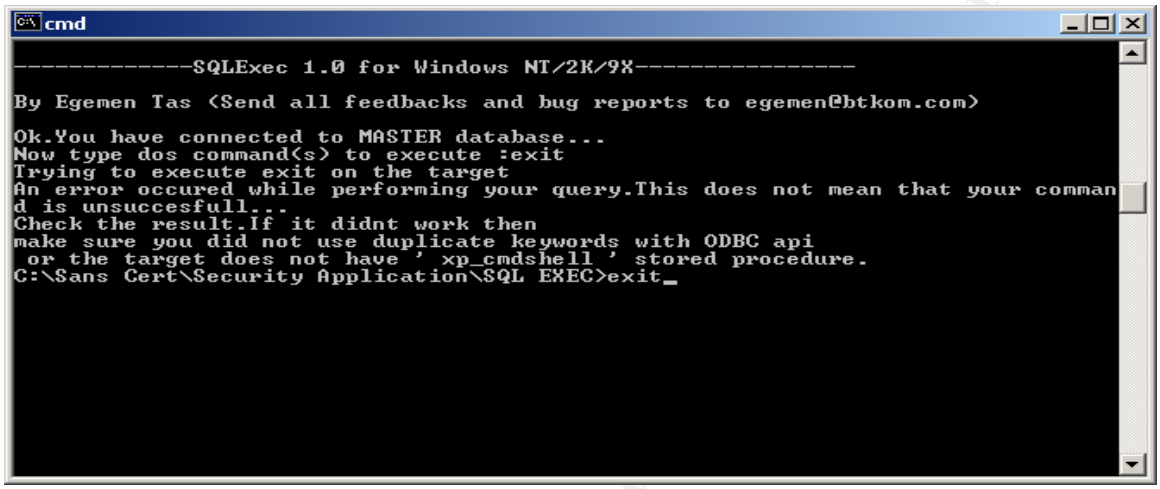
10:55:26.990027 USERMACHINE.1952 > SQLSERVER.1433: P 198:266(68) ack 405 win 17116 (DF)

10:55:27.140073 SQLSERVER.1433 > USERMACHINE.1952: . ack 266 win 8495 (DF)

10:55:27.452052 SQLSERVER.1433 > USERMACHINE.1952: . 405:1865(1460) ack 266 win 8495 (DF)

Looking at the output you will notice the three-way handshake that gets initiated. You can also see the SQL server communicating and listening on port 1433.

Once connected you can see by the screen shot below that the user is now able to type in a batch command, isql queries, scripts, etc to run against the server.



HOW TO USE THE EXPLOIT

The SQLEXEC program is run from a command prompt. Simply type in SQLEXEC and the SQL server name and it will connect.

To modify the program, locate the SQLEXEC.c file in the same directory that the SQLEXEC program is in, and open it in any text editor. Once opened you can change the login ID you wish to try and login to SQL as. You can also change the password you are trying to login as if you are attempting a brute force attack, or know the SQL server password.

Note: you must recompile the program if you change it.

This exploit can also be used in conjuction with many other vulnerablities in Windows NT and SQL. For example, Microsoft has posted several security bulletin's dealing with IIS and MDAC drivers. Many times, there are ODBC drivers created on the IIS servers that connect to the SQL servers. There are also vulnerabilities with SQL installations where the "sa" password is not encrypted and stored in plain text in the SQL.log file. This log file is kept in the "temp" directory on the SQL server and on workstations where the SQL client has been installed. This can pose a serious threat for internal hacks against a SQL server. The user just has to open the sql installation log in any text editor and he or she will see the "sa" password in clear text. Once the user

obtains the "sa" password, he or she can plug it into the SQL Exec program and they are connected.

Once you are connected you can run scripts or if you have ISQL installed you can run SQL queries or even Batch files can be run. Each of these methods can do serious harm or gather a lot of information from the SQL server, remember you are connected to the Master Database. This database holds login information, and links to all other databases, plus much more information. An unauthorized user can create and run queries, scripts or batch files against this database to obtain information on other databases, or gather user Id and password information. Once this information is obtained the options are pretty much limit less. A smart hacker will then use the other login ID's to prevent he or she from being tracked. This in turn will make traking down or even noticing something is wrong a huge task for the DBA or system administrator.

SIGNATURE OF ATTACK

Below is a sample TCPDUMP output of a SQL Server connection thru Enterprise manager. If you compare this output with the output from above, unfortunetaly you will notice that there really is not much of a difference.

11:09:13.801370 USERMACHINE.1973 > SQLSERVER.1433: P 2677082155:2677082197(42) ack 1926648352 win 17225 (DF)

11:09:13.803166 SQLSERVER.1433 > USERMACHINE.1973: P 1:167(166) ack 42 win 8154 (DF)

11:09:13.812372 USERMACHINE.1973 > SQLSERVER.1433: P 42:110(68) ack 167 win 17059 (DF)

11:09:13.813274 SQLSERVER.1433 > USERMACHINE.1973: P 167:184(17) ack 110 win 8086 (DF)

11:09:13.813357 USERMACHINE.1973 > SQLSERVER.1433: P 110:138(28) ack 184 win 17042 (DF)

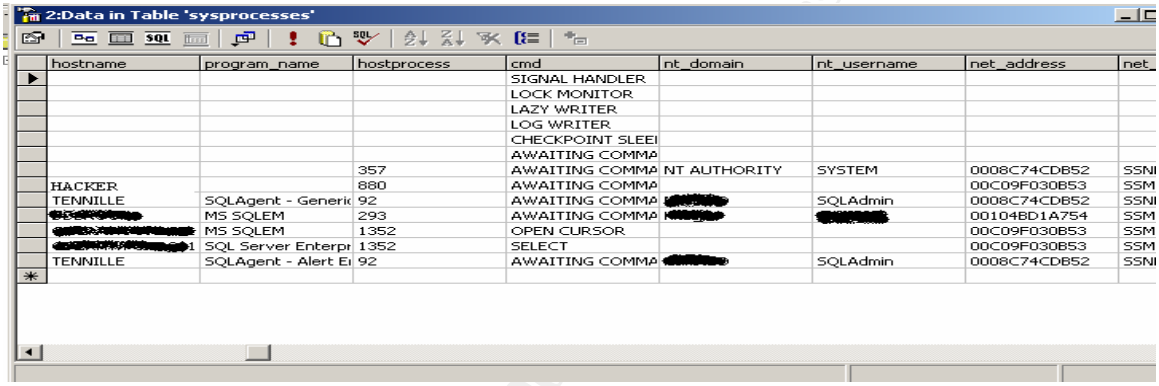
11:09:13.814806 SQLSERVER.1433 > USERMACHINE.1973: P 184:336(152) ack 138 win 8058 (DF)

To really illustrate this exploit, I used Microsofts SQL Enterpise Manager and SQL Query Analyzer. Both of these applications will show certain abnormalities with the connection thru SQL Exec.

Using SQL Enterprise Manager, you can open the master database of the SQL Server. Then go into the sysprocesses table. This table shows all users, host ID's, spids, etc of all

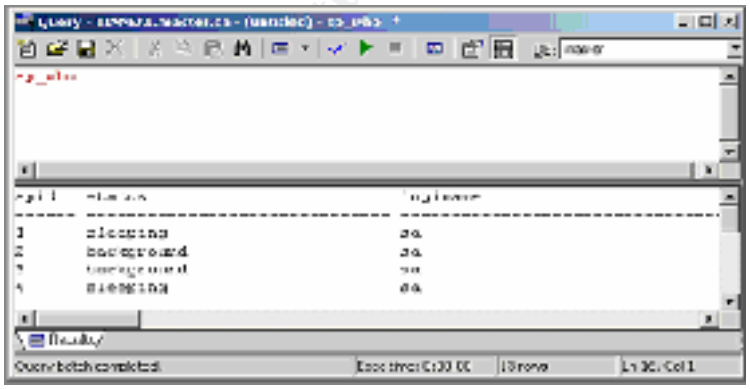
current activity on the server. From here you can see all users who are connected with the "sa" login ID.

There are some assumptions that have to be made when using this technique. The first one being that only the "sa" users are on the internal network. When you are looking at the sysprocesses table and you notice a user is logged in with the "sa" login but no computer is registered or an invalid computer name is there, alarms should start to go up. I am not saying you should start to panic, but I would seriously start looking to where this person is logged in at and why.



hostname	program_name	hostprocess	cmd	nt_domain	nt_username	net_address	net_
			SIGNAL HANDLER				
			LOCK MONITOR				
			LAZY WRITER				
			LOG WRITER				
			CHECKPOINT SLEEI				
			AWAITING COMMA				
		357	AWAITING COMMA	NT AUTHORITY	SYSTEM	0008C74CDB52	SSNI
HACKER		880	AWAITING COMMA			00C09F030B53	SSM
TENNILLE	SQLAgent - Generic	92	AWAITING COMMA	[redacted]	SQLAdmin	0008C74CDB52	SSNI
[redacted]	MS SQLEM	293	AWAITING COMMA	[redacted]	[redacted]	00104BD1A754	SSM
[redacted]	MS SQLEM	1352	OPEN CURSOR			00C09F030B53	SSM
[redacted]1	SQL Server Enterpr	1352	SELECT			00C09F030B53	SSM
TENNILLE	SQLAgent - Alert E	92	AWAITING COMMA	[redacted]	SQLAdmin	0008C74CDB52	SSNI

Another way to track this exploit and probably the most common way is to run query analyzer, specifically the sp_who and sp_who2 stored procedures. Both of these will illustrate who is logged on and where. They will show other valuable information such as the MAC address of the remote machine, host name, and in some cases the user name, Domain Name and what is currently running from that Spid.

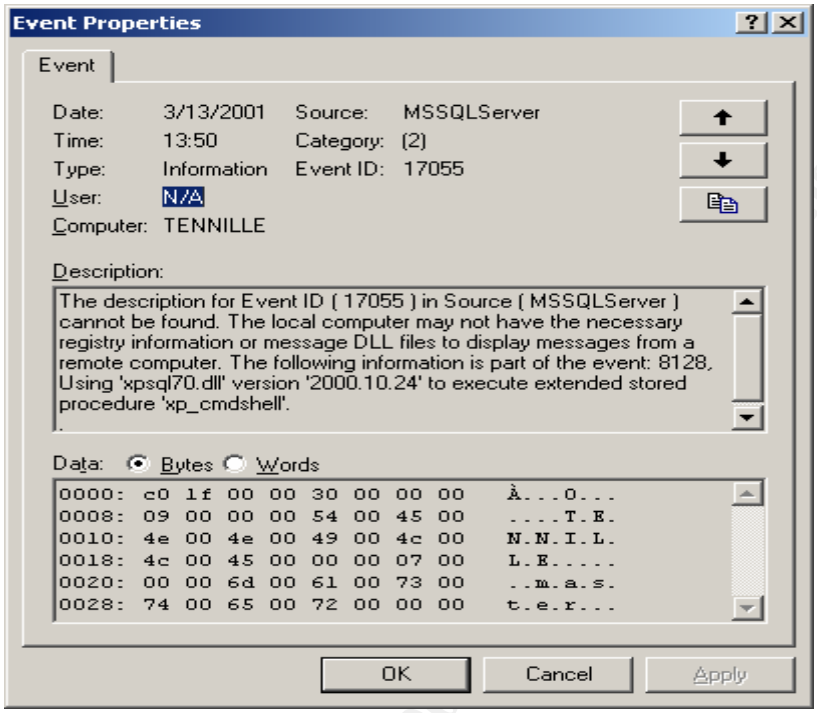


It is good common practice to run these stored procedures a couple of times a day to see what exactly is going on your SQL servers. It is also a good to keep records and create a baseline for activity.

This exploit can also be registered in the SQL server's Event Viewer. I say "can" because it only appears to generate a message if a command is typed wrong from the

SQL EXEC shell. You can see from this snapshot, the error's involving the xp_cmdshell and the remote computer. Unfortunetely the Event ID is not specific to this error. Setting a filter on this event ID will not work, unless you don't mind reading several other error's.

Here is a screen shot of what you can look for:



HOW TO PROTECT AGAINST THIS ATTACK

There are several ways to help protect against this exploit. First and probably foremost, follow common practice. Assign a password to the "sa" login, only the SQL DBA and System Administrator's need to know this password. Network Administrator's do NOT need to know this. Limit ODBC driver connectivity and enforce authentication while using them. Change the default database for the "sa" ID. I suggest changing it to the Pubs database. Not only for security reasons, but if mistakes are made with the "sa" login ID they do not take place on the Master database. Don't have server names reflect what kind of application is running them. For example you do not want a server named SQLServer.

The other vulnerabilities mentioned have patches available at Microsofts website.

Microsoft Security Bulletin (MS99-025) July 17, 1998
http://www.microsoft.com/technet/security/bulletin/fq99-025.asp

Microsoft Security Bulletin (MS99-025) July 17, 1998 Revised July 23, 1999
http://www.microsoft.com/technet/security/bulletin/ms99-025.asp

Microsoft Security Bulletin (MS00-014) March 8, 2000
http://www.microsoft.com/technet/security/bulletin/ms00-014.asp

---

SQL EXEC SOURCE CODE

```
{
    SQLCHAR Host[512]="";
    SQLCHAR *User="";UID=sa";
    SQLCHAR *Pass="";PWD=";
    SQLCHAR *Database="";
    SQLCHAR InConnectionString[1025]="";
    SQLCHAR rowBuff[200]="";
    SQLINTEGER   iRowBuff;
    UCHAR Query[1500]="";
    UCHAR Cmd[300]="";
    char inBuff[1025]="";
    SQLRETURN nResult;
            printf("\n\n-------------SQLExec 1.0 for Windows NT/2K/9X-------------
\n\nBy Egemen Tas (Send all feedbacks and bug reports to egemen@btkom.com)\n\n");
          printf("\nUsage : SQLExec <Hostname> \n!!!!(Do not use ip addresses of
targets)!!!!\n");
    …….
```

Listed above were some excerpts from the SQL EXEC code. You can see how easy it would be to change the code to fit the hackers needs.

You will notice that the Database field is blank. This is because on a default installation the “sa” accounts default database is the Master. Once again, if a hacker

knew enough about the SQL server, he or she could manipulate this program and taylor it to their needs.

The bottom line of the code clearly states not to enter in an IP address. This is because SQL server communicates thru host names either thru WINS or DNS *gethostbyname* requests.

ADDITIONAL INFORMATION RESOURCES AND REFERENCES

Microsoft Security Bulletin (MS99-025) July 17, 1998
http://www.microsoft.com/technet/security/bulletin/fq99-025.asp

Microsoft Security Bulletin (MS99-025) July 17, 1998 Revised July 23, 1999
http://www.microsoft.com/technet/security/bulletin/ms99-025.asp

Microsoft Security Bulletin (MS00-014) March 8, 2000
http://www.microsoft.com/technet/security/bulletin/ms00-014.asp

http://www.cisco.matc.net/sem2v2.0

http://packetstorm.securify.com/filedesc/SQLExec.zip.html

http://www.microsoft.com/TechNet/security/bulletin/MS00-035.asp

Comer, Douglas [1999] Computer Networks and Internets, Prentice Hall Upper Saddle River, New Jersey 07458

Talmage, Ron [1999] Microsoft SQL Server 7 Administrator's Guide , Prima Tech Publishing Rocklin, California 95677

Stevens, W. Richards [1995] TCP/IP Illustrated, Volume 1, Addison-Wesley Publishing Company Reading, Massachusetts 01867