GIAC Certified Incident Handler (GCIH)
Practical Assignment
Version 2.1 (revised April 8, 2002)

Assignment track: *Support for the Cyber Defense Initiative*

Assignment issue:

Attacks upon Microsoft Database Server
(Analysis of vulnerability and beyond)

Prepared by: Yaakov Bezalel

Completed on the 17 January 2005 (This assignment was pre-approved)

Preface:

During the years 2000, 2001 and 2002 there have been two major phenomena occurring until this very t ime.

The first phenomenon is that organizations, parties and individual people relay more and more on computing resources that are available via the Internet. People seem to have less and less idea of what happens in their computer and network while they use the internet - that multitude network of computers tied together and allowing people to do day -to-day operations without distinction how it is done and which computer in the network is involved in the activity. Even the most knowledgeable people in th e computing arena, have lack of knowledge for at least part of what happens while working in the internet. This is mainly due to the increasing complexity of the components of the computing environment.

The second phenomenon is that attacks grew in number s and in the amount of damage they carry.

Both phenomena were proved true, as the amount of damage due to those attacks have been growing and estimated in billions of dollars world wide.

Many of the latest attacks were directed to hit and also continue and spread through the very foundation of the Internet – the **Web Servers** .
A Web Server is an application, a program that is written to receive and to respond for requests generated by people surfing the internet or using the internet for any other communic ation.
In the past few years there have been two such Web Server applications used in the Internet infrastructure, Microsoft IIS (http://www.microsoft.com) and Apache Software Foundation (http://www.apache.org).

Most of the attention to actual and possible attacks is still directed to the Web Servers. Still, attacks are coming at the Database Servers as well.

What is a Database server ?

A Database server is an application, a complex set of programs set to assist in the organization of important and meaningful information within the computer system.
Most of day to day operations that involve data entry are actually sent by the specific application that you use, to the Database Server.
The applications that you use could be those inside your organization or provided by a company over the Internet.
So when you file a Request for a new Driver License, when you file a Purchase order, when you buy a Burger in your favorite Store, in almost all those cases, the details entered into the Computer System are being sent to the Database Server (details such as your Credit Number, Age, Item to Purchase and so forth).

Most of the critical and worthy data pieces shall be stored on the Databa se server. So the Database Server is a very important and actually defined as the MOST CRITICAL application within the computer system.

During the past months, two of the Major commercial Database Server manufactures, Microsoft (http://www.microsoft.com/security) and Oracle (http://otn.oracle.com/deploy/security/alerts.htm) have published critical vulnerabilities within their Database pr oducts, that could allow attackers to gain access from within their remote computer and into the Database Server computer, and also gain high access permission into the Database information units as well.

In this paper we'll discuss one of the vulnerabili ties and how to protect your Database.

How does a Database Server provide it's services?

As stated a Database server is a program, a complex one, which waits for requests for data processing.
In the Microsoft environment such programs that provide service s for requests are referred as **"service" or "services".**
The requests shall be of other programs that people run – let's call them **"client programs".** In many cases the Client programs are ran on a computer which is not the one that the Database Server runs on.

So many times the "Client programs" have to communicate with the Database Server, over the network.
How is that possible?

Actually, many other programs, such as your Email program or Web Browser communicates with other computers, to be able and fulfi ll your requests, be it read an email, or look at a site on the web, or get software updates.

All the communications between programs are done over some kind of physical and electronically equipment (that is used to pass over the information) that is wire s, modems, satellites and so forth. We shall not dwell more into that Arena.

However, the programs have to send the data (Your name or Credit Card ID or Web address that you want to visit, etc.), using a format that the receiving programs (servers such as Database Servers) will surely understand.
There are several basic networking data formats. The most commonly used is called TCP/IP ("Transmission Control Protocol/Internet Protocol"). The TCP/IP format uses two basic ways of communication: **UDP** ("User Datagram Protocol") and **TCP** ("Transmission Control Protocol").
The TCP/IP is implemented as a set of programs within each computer that handle the data sending and receiving.
There is also a special infrastructure of Communication cards within the computers that are well adjusted to forward the data from the TCP/IP programs, and transform the data into a signal that can be physically sent Through communication extension devices until they reach their destiny, on the receiving Communication card of the target co mputer. The target machine's Communication card transforms the signals into data within the computer memory for the TCP/IP programs to process and send to the Database service program or other service programs.

Any communication sent, using TCP/IP, is bro ken down by the special Networking programs (within the "sending" computer) into small pieces (called packets) and sent to the receiving computer, where it's re -structured into a whole again and put for the Database Server program (service) for processing.
The Database service will communicate back to the "Client Program" with a response and sometimes with Actual data sent back (the results of a request to give a report of last purchases, the confirmation of Money deposits, etc.).

This operation of break -down and re-structuring, maintains data in transit, during processing within the Computer memory.

As said, the TCP/IP communication system has two basic methods of communication protocols.

The TCP protocol is considered a more reliable method since it keep s track of information units and sees that all the parts of your communication are delivered completely and in the original order they were originally compound.

The UPD protocol is considered a less reliable one, as the programs handling its delivery, do not make sure that all the packets have arrived and in the original order. In this case communication may be incomplete, yet, since the UPD programs do not have to "worry" about order and reliability, it takes them less time to process data, compared to TC P.

So – do you loose data using "UDP"?
Not necessarily, since the "Client programs" and "Server Services" will take care of reliability, where using TCP protocol they do not have to do so.

Some applications or programs do not handle data communication re liability and therefore have to use TCP for critical data delivery.
Some programs can use TCP or they can use UDP (and for UDP they make sure data is communicated fine) which makes more sense if they can do it better that the TCP mechanism.
So UDP program s can handle reliability faster than TCP mechanism does since TCP carries a very complex and extensive mechanism to handle data verification and recovery. The TCP intentional purpose was to allow data to travel via a very long distance, hoping through many machines and communication devices that may fail during delivery.

The "Client Programs" can complete UDP lack of reliability handling by implementing only part of the original TCP reliability requirements, assuming they ran in a basically simple and reli able environment.
Such reliable environment could be a network within a building rather between two countries.

In some cases "Client Programs" run on the same machine as the "Database Server or Service" is located.
The communication in this case, can sti ll be based on any networking protocol as used between two computers, sicne the different programs within the computer are able to send information from the machine to "itself", putting the data each time either for the "Database service" to process or for the "Client Programs" to get a response.

There are databases and other Server programs that could be operated within the internal computer without any communication done outside of the machine for those services purpose. However this is becoming more and rarer.

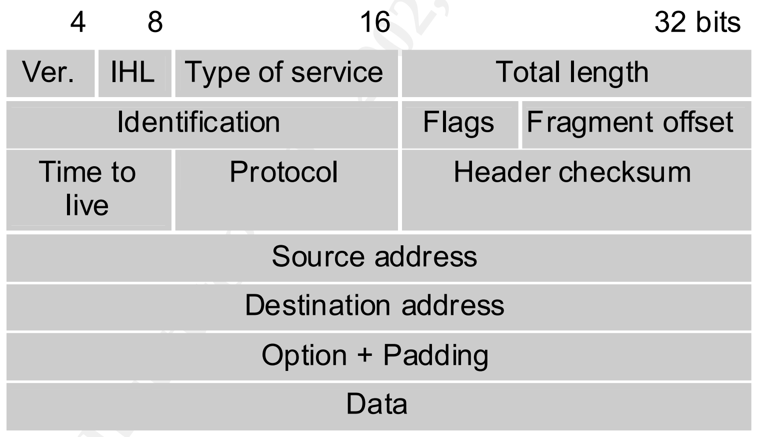
The mainframe computers as well, provide more and more of their Server Services available for Client Programs over the network.

What are the parts of TCP/IP data item (packet)?

We shall not dwell into this arena too much, as there are many other resources explaining deeply the structure of networking, for example at http://www.protocols.com/pbook/tcpip.htm
Still, to enable understanding of the material discussed in this document, we'll browse through several items that are part of TCP/IP.

In essential the TCP/IP data item that is transmitted (a packet) is constructed from several components, hereby a diagram from http://www.protocols.com/pbook/tcpip.htm#TCP
You may look there for additional information about TCP/IP and its items that were not fully discussed here.

Basically each data item is divided into packets by the TCP/IP networking programs and then sent away.
In our conversation we'll discuss IP, TCP and UDP.
The TCP/IP packet has IP part of Header with addressing related information (who wants to send the data, to whom, how) and Data to send.
Then there is the TCP or UDP information implanted within the IP data section.



IP header structure

From the site http://www.protocols.com/pbook/tcpip.htm#TCP

The most important parts for our discussion, in the IP header are:

1. Source address (Source IP address) – who sends the information, so if problems occur we know whom to notify about it.
2. Destination address (Destination IP Address) – to whom is the data sent, so networking devices and computers along the way, can know where to further route the information.
3. Total length, so TCP/IP programs can verify all data has been actually transmitted.

4. Fragment offset – since packets may be split into several parts to allow passing through relatively slow and low -resource networking devices, the packet break-down parts are numbered and called fragments.

5. Data Section: The IP data section includes the specific TCP or UDP parts that have header and data sections of their own:

							16	32 bits
Source port								Destination port
Sequence number								
Acknowledgement number								
Offset	Resrvd	U	A	P	R	S	F	Window
Checksum								Urgent pointer
Option + Padding								
Data								

TCP header structure

From http://www.protocols.com/pbook/tcpip.htm#TCP

Let's get familiar with the major items in the TCP Data header.

What is a "port" ?

In most computers there are many types of "Computer Programs" each one needs to communicate at each moment with specific "Server Programs or services".
Since all the "Client Program" requests may take time to handle, meaning a computer can not handle all requests immediately, a queue mechanism is built into computers and serving many programs. The Networking programs have their queues as well.
The people who built TCP/IP and defined the requirements of the TCP/IP service and client programs, tried to make it easier for service applications to handle only their requests from the many waiting at the queue, so they defined the term "port". Most of service programs that need to communicate are defined with a public known TCP/IP port. **The port is a kind of a label that the service programs are listening for and will look ONLY into communication packets that have the appropriate port number.**
Sometimes during a communication between a Client program and a Server service, they "agree" of a new port to further c ontinue the communication.
In any case, the "Source port" is the port that the "Client application" was using to enter its request for queue sending, while the "Target port" is the port number that the "Server service" is listening for, within another comp uter, or the same computer that the "Client program" is running on.

Looking at both the IP and TCP headers we can summarize:

You could look at **ports as if they were Zip -Code** of regular day-to-day mail that you put in a mailbox where it's delivered to th e post office and from their sent to the home of the recipient. The Zip -Code specifies the area of the recipient rather than the actual person it's delivered for.

The **computer Source and Destination IP address** has several parts: Network ID number (Country , City and Street) and Machine IP number (the mailbox with name label) where all mailboxes are serving the same building but for different people living there.

The TCP/IP (known as IPv4 as well) address is compound of 4 dotted numbers such as 192.168.12.1 00.
There is a new IP protocol starting to be deployed which is called **IPv6** and it allows more reliable secure communication and to a much mush larger addresses.

The whole Networking scheme has several layers, each one providing other services, be it hand ling physical connections, securing the communication, handling the data processing (break -down and re-build), providing identification of machines and devices and more - Some of them I mentioned earlier, for the others you can look at the references for T CP/IP.

The **Microsoft Database server** (**seldom referred to as "MS -SQL" - service**) can communicate using TCP/IP communication protocol. It allows both TCP and UDP communication.
The MS-SQL services are compound of several Service -programs that **listen for Client-Programs in the TCP/IP port 1433.**
The networking programs bundled with MS -SQL can receive their packets using at least 2 methods: Sockets and Named Pipes.

The Names Pipes works in most cases using TCP/IP as an underlying mechanism for communication, t hough it can use other protocols as well. The named pipes are data structures, maintained by the Microsoft NT/2000 operating system in such a way that you can have programs send a message or data to another program, while both use a very simple standard me thod to listen for data and to send data. The Client and Server programs do not have to worry about how naming of those Named Pipe "mailbox -like" are maintained, nor how data will travel to each side.

You can watch your current named pipes that applicatio ns use, in the Microsoft Operating system network connections viewers.
One of the viewers is part of the MS -SQL version 7.0 and called **"Server Network Utility"** and it can show you named pipes such as: **'\\.\pipe\sql\query.'**
You can read more about Named Pip es at Microsoft TechNet:
http://msdn.microsoft.com/library/default.asp?url=/library/en -us/rpc/ov-pipes_1pv7.asp

The sockets mechanism is implemented us ing a "sockets library" that holds all basic operations to maintain Data Sending between applications, and is based on socket identification numbers both on the listener program and on the sending data program.
The "sockets library" is a standard set of pr e-prepared small programs that are supplied on most operating systems by the Computer and/or operating system vendor. Those programs handle all the operations to insure listening and sending operations and availability of socket resources acquired for the client programs that need them. In Microsoft Operating systems this mostly referred as the **"Winsock"** API, Winsock DLLs and other components.

For more information on Windows sockets you can try resources such as:
The book "Windows Sockets Network Programmi ng"
Written by Bob Quinn and Dave Shute, with foreword by Martin Hall
at http://www.sockets.com/toc.htm (shipped at late November, 1995)

You can watch current opened sockets using the Microsoft **"netstat -a"** NT command.

The MS-SQL Server can communicate using other methods than TCP/IP TCP and UDP. This includes Novell Netware, Apple AppleTalk and VINES.

Looking at the MS -SQL version 7.0 Database server services (SQL 2000 is similar in this aspect), it is imp lemented as several Server Programs running in the Microsoft NT and Windows 2000 platform (you can look at them on the Windows Control Panel "Services" option):

<u>The SQL Server services:</u>

Looking at the Microsoft "SQL Server 7.0 Books Online" we can learn the definitions of each of the Services. I have put the Microsoft Books online text in *Italic* formatting and put some numbering of the original text.

"….

1. **MSSQLServer** – Implemented as An NT Service or a running program "*SQL Server manages all of the files that comprise the databases on the server. It is the component that processes all Transact -SQL statements sent from SQL Server client applications. SQL Server can also execute stored procedures in other remote servers and supports distributed queries that retrieve data from multiple sources, not just SQL Server*."
2. **Distributed Transaction Coordinator** - manages Distributed Transactions (or Distributed MS -SQL Client Program Requests). Distributed Transactions " *are transactions that involve resources from two or more sources. Microsoft® SQL Server™ supports distributed transactions, allowing users to create transactions that update multiple SQL Server databases and other sources of data* ."
3. **MS SQL ServerAgent** – "…
   SQL Server Agent supports features allowing the sc heduling of periodic activities on Microsoft® SQL Server™, or the notification to system administrators of problems that have occurred with the server. The SQL Server Agent components that implement this capability are:
   a. *Jobs - Defined objects consisting o f a one or more steps to be performed. The steps are Transact -SQL statements that can be executed. Jobs can be scheduled, for example, to execute at specific times or recurring intervals.*
   b. *Alerts - Actions to be taken when specific events occur, such as a specific error, errors of certain severities, or a database reaching a defined limit of free space available. The alert can be defined to take such actions as sending an e -mail, paging an operator, or running a job to address the problem.*
   c. *Operators - People identified through their network account or e - mail ID who can address problems with the server. They can be the targets of alerts, either through e -mail, a pager, or a net send network command…."*

…"

To further show the available possibilities for Datab ase Automatic hostile operations, take a look at what the MS SQL ServerAgent service is capable to handle in Operations Automation and Imagine what happens if unauthorized attacker enters the MS -SQL database and sets a Job, Alert or Operator to serve he's needs.

More from Microsoft "SQL Server 7.0 Books Online":

> *"…*
> *Jobs, alerts, and operators are specified using:*
>
> i. *SQL Server Enterprise Manager GUI.*
> ii. *Applications that use SQL Distributed Management Objects (SQL-DMO).*
> iii. *Applications that use Transact -SQL and a standard database API.*
> iv. *The definitions are stored by SQL Server in the msdb system database.*
> v. *When the SQLServerAgent service is started, it queries the system tables in the msdb database to determine what jobs and alerts to enable.*
> vi. *SQL Server Agent e xecutes jobs at their scheduled time.*
> vii. *SQL Server passes any events that occur to the SQL Server Agent.*
> viii. *SQL Server Agent executes any alerts, or sends SQL Mail requests to SQL Server, or sends net send commands to Windows.*
> ix. *SQL Server version 7.0 is more highly automated than earlier versions of SQL Server, and does a better job of configuring itself automatically to meet processing demands. These features lower the potential for exception conditions that would trigger alerts. Scheduled jobs remain a good feature for implementing recurring tasks such as backup procedures.*
>
> *…"*

You can read more about MS -SQL Server at
http://www.microsoft.com/sql/default.asp

The SQL (MS-SQL included) protocol:

The client programs and the Server programs communicate requests and responses on behalf of their operator. It is using a networking protocol to send the requests as TCP/UDP data; Still this TCP/UDP data is actually also interpreted by MS-SQL Client and Server programs as a protocol of itself.
So the TCP/IP protocol handles several layers of the complete networking layers model "OSI layers model" (you can take a look at http://www.protocols.com/pbook/tcpip.htm#TCP).
Still as the network packets reach the MS -SQL server service, it requires parsing of the data as actual SQL requests – this is called the Application level in the OSI layers model.

The SQL language ("Structured Query Language") consists of a set of operations that allow programs to query information and process it while working on a Database that is organized in **"Relational Structure or (also called "Relational Database").**

A relational Database includes many items, still the basic objects it uses are:

1. **Columns** – include single item information such as "Name", "Age", "Amount of money"
2. **Rows** – A collection of Columns that relate to the same specific instance they describe, such as Fields of ID, Name, Age and Address are Columns of specific "Citizen Details"
3. **Tables** – A collection of rows that relate to the same set of issues. For example the complete list of rows containing "Citizen Details" are grouped and attached with a name "Citizens Table".

The SQL language consists of two basic operation types:
DML and DDL.

Both DML and DDL examples are similar to those at The W3Schools web site, owned by "Refsnes Data" company at:
http://www.w3schools.com/sql/sql_intro.asp and are in *Italics*.

The DDL ("Data Definition Language") allows operations that include management of Database objects such as Deleting a table (DROP TABLE), create a COLUMN (CREATE COLUMN) and operational issues such as Starting a backup dump of the Database or Shutting it down.

CREATE TABLE Customer
(FamilyName varchar(90),
PrivateName varchar(90),
Address varchar(250),
AccountNumber integer)

DROP TABLE Customer

The DML ("Data Manipulation Language") allows operations that include data Query (SELECT), adding item s (INSERT), removing items (DELETE) and updating contents of a current item (UPDATE).

```
SELECT * FROM Customer WHERE ADDRESS LIKE '%USA%'

INSERT INTO Customer
VALUES ('Bill', 'Graham',
            '118 BlueHighway road CA USA',
            1123333 )

UPDATE Customer S ET PrivateName = 'William'
WHERE AccountNumber =  1123333

DELETE FROM Customer WHERE LastName = 'Graham'
```

As SQL requests and responses are carried on between applications, we can call it the **"SQL protocol".**

You can read more at sites such as web site of W3schools web site: http://www.w3schools.com/sql/sql_intro.asp

Focus in this paper: exploring the MS -SQL vulnerability
<u>"Microsoft SQL Query Method Vulnerability"</u>

I chose to discuss MS -SQL vulnerability since MS-SQL is one of the most deployed Database Servers as a product on its own, and within many other Software products that require a Database for their processing.
Also Microsoft in its own products and many of their most critical Infrastructure-related products, uses in many cases MS -SQL as the main repository as well as many other Software vendors. Therefore an attack on MS-SQL related services will find many targets and when succeeded it will have impact on a wide range of services within companies.

I have selected to explore a specific MS -SQL vulnerability (labeled as "CVE-2001-0344" in the "Common vulnerabilities and exposures" archives).
The reason for this selection is since it's relatively new (June 2001), actually, therefore many chances people did not take the steps to eliminate this vulnerability yet.
Also this vulnerability exists both on MS -SQL Server version 7.0 and MS -SQL version 2000 Server Gold, that makes the most recent and popular MS -SQL versions, again more of a chance p eople use it and may have vulnerable systems.

How is an attack on Database different from the ultimate attack on the Operating system?

First, let's take a look at the current automated malicious mechanisms – they ultimately try to control the Operating s ystem, the mechanism that manages all the programs and activities on a computer.

Many vulnerabilities, actually allow the attacker to break the Client Program communication to the Server service in such a way that they gain larger privileges on the attack ed Computer, such that they can further deepen their hold of the system and ultimately send it with the most power -full instructions using **"Operating System Command Shell"** that is the interface that allows most of the major control operations for a compute r system.

So it seems that the most ultimate mission of an attacker, as he is provided with "Command shell" or similar power -full access to the Victim computer, is to easily instruct the attacked computer to create a more hidden environment in which the a ttacker can hide it's entrance, erasing all traces in the system logs and files, and installing special programs to hide an easy way into the computer. As this occurs, the attacker can get back in without having to use the original exploit for the vulnerab ility of the system.
Those special programs operated by the attacker are seldom called **"root-kits". Those programs hide the existence of the attacker operations** from the Computer monitoring programs, allowing the attacker to continue its ride into Data Compromise without a rush.

This is Pretty nit – right?

In view of all this, the MS -SQL vulnerability may seem less -important – it does not necessarily allow access to the "Command Shell", it basically just allows you to do almost whatever you want within th e Database.

Most people that do not know what a Database contains would rather handle other types of obvious threats.

However, a database (including MS -SQL) can allow many damaging operations including getting an Operating system Command Shell and then, almost any application could be compromised.

In some cases the breakout from the Database into the Operating system could be done using a high permission Operating System level, gained from the MS-SQL compromised services.

How is the "Microsoft SQL Query Method Vulnerability" possible?

The nature of this Vulnerability requires that an attacker that will send a request to the MS -SQL Server service (this request is called SQL Query), using a specific SQL request type (called **SQL Query Method**).

If the Database Server uses an authorization **method called "Mixed Mode",** While the attacker coordinates an attack close to the point in time at which a login to Database Administrator account was done, the attacker will login using a **local** MS-SQL regular (non -privileged) account and then he will be able to gain the highest privileged operation mode and operate within that mode, by operating a specific vulnerable SQL method available within the Database.

The vulnerable MS -SQL method does not check who originally requ ested it's operation, instead it runs the operation under a previous session that had high level of permissions, allowing an attacker to ask the SQL Method to do SQL operations that will later allow him to enlarge he's break -in.
Further operations done on behalf of that local user will allow the attacker to do hostile and very damaging operations.

Let review the login process and the meaning of "Mixed Mode"?
When you want to access a Database you will need to ask for permission from the Database.
This process of permission request is called many times **"login"** or "**connection**" or "**Authentication**", during which you are requested as a pop - up prompt in your screen, to enter a **"user"** name that should have been supplied to you only. You also enter a secret word – **"password"** - associated with that "User Name" that only you should have known.
This allows the assumption that you are the only person using this "user" information. **Based on that, the Database Server programs allow Client - programs that you run to do actua l operations within the Database.**

Many times this login process is noticed only when you start -up your computer and use a User and Password on your initial Operating System connection, having all the next programs and Database access validated against this initial User and Password entry rights that you received.

A "Local" database user refers to an existence of a user and password within the Database. Other methods of a **non-local user in MS -SQL** refer mainly to situations in which the person has done logi n to the Computer Operating system rather than the Database. However the Database may be configured to identify this login and map the Operating System user to a set of permissions this user should have in the Database.
Some databases will only accept "Loc al" users login, some may allow only non-local logins (called NT Authentication in MS -SQL), and some may allow both – **which is "Mixed Mode".**

More information about this vulnerability can be found at those web sites:

1. The Common vulnerabilities and expo sures site: <http://cve.mitre.org/cgi -bin/cvename.cgi?name=CVE -2001-0344>
2. The Microsoft Corporation Technet site: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS01-032.asp>
3. The Computer Incident Advisory Capability site http://www.ciac.org/ciac/bulletins/l -095.shtml

Vulnerabilities Associates with the MS -SQL Database service port 1433:

Looking at the Internet Storm Center web site
< http://isc.incidents.org/port_details.html?port=1433 >
We can browse the following table with latest MS_SQL Database server, port 1433 related vulnerabilities.

For any **CVE ID** mentioned, you can use this URL http://cve.mitre.o rg/cve/
To get the all details including reference to the Vendor(s) site of the product(s) discussed in the vulnerability issue.
The Microsoft vulnerabilities site is located at:
<http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security /bulletin>
As we shall focus only on one of the Vulnerabilities, I will add a brief explanation for each of the mentioned common vulnerabilities.
The orig inal content in the table above was copied from the original web site (< http://isc.incidents.org/port_details.html?port=1433 >) and appears in *Italics*, while my additions are in normal text.

Vulnerabilities for this port (from CVE)

CVE ID	***Protocol***	***Source Port***	***Targetport***
Description			
CVE-2001-0344	*tcp*	*Any*	*1433*
An SQL query method in Microsoft SQL Server 2000 Gold and 7.0 using Mixed Mode allows local database users to gain privileges b y reusing a cached connection of the sa administrator account. This vulnerability shall be thoroughly discussed later in this paper.			
CVE-2000-0603	*Tcp*	*Any*	*1433*
Microsoft SQL Server 7.0 allows a local user to bypass permissions for stored procedures by re ferencing them via a temporary stored procedure, aka the "Stored Procedure Permissions" vulnerability. Stored procedures are SQL statements that can be inserted into a Database, given a known name that can be used to operate them by other programs and other Database stored procedures. The stored procedures as any other database objects have permissions, allowing only allowed database users to use them. This vulnerability actually exposes the Database to usage of temporary ad -hoc created hostile procedures t hat can override the basic permission mechanism and possibly add to the database, a hostile code. More information at <http://www.microsoft.co m/technet/treeview/default.asp?url=/TechNet/securit y/bulletin/MS00-048.asp>			

CVE-2000-0485	*tcp*	*Any*	*1433*
Microsoft SQL Server allows local users to obtain database passwords via the Data Transformation Service (DTS) package Properties dialog, aka the "DTS Password" vulnerability. The DTS is an option within MS -SQL server that allows creating procedures (explained above) that could be scheduled to run automatically. During the creation of such procedures, a Database User and Password is typed by the procedure creator. It was found that is it relatively easy to find the password using the MS-SQL database management application called "SQL Server Enterprise Manager". More info at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS00-041.asp>			

CVE-2000-0402	*tcp*	*any*	*1433*
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. The installation of MS -SQL service packs 1, 2 and 3 may leave under certain conditions installation log files . Those files may include the password of the most privileged user that can do anything on the database (that is the "sa" administrator user). As the attacker views the sa user's password he can then login to the database and compromise it. Since the defau lt place those installation log files are located at has default very un -restrictive permissions, there is a big chance for attackers to view them. More information can be found at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS00-035.asp>			

CVE-2000-0202	*tcp*	*Any*	*1433*
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. This vulnerability allows under certain conditions, that people who are allowed to send SQL Queries to MS -SQL server and MSDE (I will not discuss MSDE here and more information is availa ble in Microsoft web sites) and are able to submit certain carefully crafted illegal and un -expected SQL queries, will actually be able to carry on hostile operations within the Database and in some cases within the Operating system that hosts the MS -SQL Database server. More information can be found at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS00-014.asp>			

CVE-2000-0161	*Tcp*	*any*	*1433*
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. The Microsoft Site Server 3.0 that uses MS -SQL contains Sample Web Sites that include SQL programs to be operated via a Web Browser. Some of those Programs execute SQL statements - as a result of the user input via the Web Browser. However the vulnerable Web Server programs do not validate the input given to them by the web browser, so the hostile user could craft the information sent to the Web Server using he's web Browser, so that hostile SQL statements are added to the legitimate information that was collected on the Web Server form, for submission to the Web Server. Those hostile SQL statements will be given for execution by the vulnerable Web Site programs, to the MS -SQL server, allowing the attacker with any operation that the vulnerable Web Site programs are allowed at the MS -SQL Database. This Web Si te attack is commonly called " **SQL Piggyback**". There is a good article written as part of Incident Handler certification, by Matt Borland " Advanced SQL Command Injection: Applying defense -in-depth practices in web -enabled database applications ", dated 1 /19/2002 that can be found at: < http://www.giac.org/practical/Matt_Borland_GCIH.zip > More information is available at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS00-010.asp>			

CVE-1999-0999	*Tcp*	*any*	*1433*
Microsoft SQL 7.0 server allows a remote attacker to cause a denial of service via a malformed TDS packet. This vulnerability allows an attacker to send specially crafted packet to MS - SQL server port 1433 (if MS -SQL was configured to "listen" and respond to requests at this port), which will cause the MS -SQL server to stop servi ng requests. The MS -SQL service should then be restarted to regain normal service mode. More information can be found at: <http://www.microsof t.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS99-059.asp>			

There are some new vulnerabilities reported, most of them are similar to those mentioned above, I have listed here some of them:

CVE-CAN-2002-0641
CVE-CAN-2002-0624
CVE-CAN-2002-0642
CVE-CAN-2002-0624

Those latest vulnerabilities and patches are discussed in Microsoft security bulletin that were recently published:

"MS02-034: Cumulative Patch for SQL Server (Q316333)"
And
"MS02-035 SQL Server Installation Process May Le ave Passwords on System (Q263968)".

Exploit details:

Name:	"CVE-2001-0344" Is the ID within the " Common vulnerabilities and exposures" archives and it was also discussed in Microsoft Security bulletin MS01-032, Available at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS01-032.asp> Labeled as: **"SQL Query Method Enables Cached Administrator Connection to be Reused"**
Variants:	None found.

Operating System:

List copied from Security Focus vulnerabilities site at:
<http://online.securityfocus.com/bid/2863/info/ >

Microsoft SQL Server 7.0 SP3
- Microsoft SQL Server 7.0

Microsoft SQL Server 7.0 SP2
- Microsoft SQL Server 7.0

Microsoft SQL Server 7.0 SP1
- Microsoft SQL Server 7.0

Microsoft SQL Server 7.0
- Microsoft BackOffice 4.5
- Microsoft Windows NT 4.0
- Microsoft Windows NT 4.0 SP1
- Microsoft Windows NT 4.0 SP2
- Microsoft Windows NT 4.0 SP3
- Microsoft Windows NT 4.0 SP4
- Microsoft Windows NT 4.0 SP5
- Microsoft Windows NT 4.0 SP6
- Microsoft Windows NT 4.0 SP6a

Microsoft SQL Server 2000 SP1
- Microsoft Windows 2000 Workstation
- Microsoft Windows 2000 Workstation SP1
- Microsoft Windows 2000 Workstation SP2
- Microsoft Windows NT 4.0 SP5
- Microsoft Windows NT 4.0 SP6
- Microsoft Windows NT 4.0 SP6a

Microsoft SQL Server 2000
- Microsoft Windows 2000 Workstation
- Microsoft Windows 2000 Workstation SP1
- Microsoft Windows 2000 Workstation SP2
- Microsoft Windows NT 4.0
- Microsoft Windows NT 4.0 SP5
- Microsoft Windows NT 4.0 SP6
- Microsoft Windows NT 4.0 S P6a

Protocol/Service used by the exploit:

Any of the possible protocols that MS -SQL server allows, could be used to exercise the exploit. I have discussed the possible protocols relevant in previous chapters.
It is possible also to run the exploit on MS -SQL TCP port 1433 that is very much attacked those days.

Brief Description:

The exploit for this vulnerability allows a user that has access to a local MS - SQL account (that is user and password), to actually gain the privileges and access permissions of a previously operational SQL session of user "sa" (an administrative user account that has the highest permission level).
By running a specific SQL query method the attacker could take actions that only the "sa" user could do, that is, the attacker could la unch an attack that could do almost anything harmful to the MS -SQL database and in some cases to run operations against the hosting operating system.

Description of variants:

None found.

Protocol Description:

This was discussed previously within this paper, in the chapter **" The SQL (MS-SQL included) protocol".**

How the exploit work and how to use the exploit:

The short description was thoroughly within the chapter in this paper, labeled '**How is the "Microsoft SQL Query Method Vulnerability" possi ble?**'

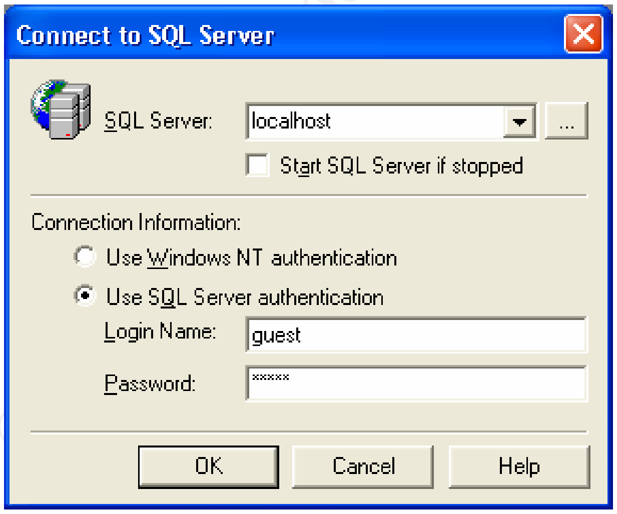
I will further dwell into it:

1. The discussed vulnerability is possible only against MS -SQL servers that are configured to use "Mixed Mode" login (or "authentication")
2. The attacker should have access on such a database, as a "local" user. Let's say the attacker knows of a local MS -SQL user called "guest" and of its password "guest". In most cases MS -SQL will accept SQL operations from other machines in the network.
   Therefore the attacker could use any tools that provide access to MS - SQL. Lets show how the attack is done using the Microsoft utility called "Query Analyzer" (provided within the MS -SQL installation).

   I will refer to button selections from the Microsoft Windows menu by the **sign "->"** meaning one should click on the button with the label nam e mentioned after the mark " ->"

   The attacker Selects "Start->Programs->Microsoft SQL Server 7.0 ->Query Analyzer and then select in the Query analyzer "File ->Connection"

   The next screen appears and the attacker fills in the database server name, the local user name and password:



SQL Server: localhost

[ ] Start SQL Server if stopped

Connection Information:

( ) Use Windows NT authentication

(●) Use SQL Server authentication

Login Name: guest

Password: *****

   You should notice that the attacked should be carried on using a known local MS-SQL user and selecting the " **Use SQL Server authentication**" rather than "Use Windows NT authentication".

3. Now the attacker has a worksheet that all ows him to write down SQL commands including a very simple syntax to use the specific vulnerable SQL command that is vulnerable so it will run using permissions of a previously authenticated "sa" administrator highest privileged account.
4. Using SQL statemen ts under this privileged mode the attacker can instruct the Database to modify Security Auditing, hide it's tracks, enter malicious code, modify data, instruct the database to perform operations onto the hosting Operating System
5. The attacker would also cre ate an easy way back (for example set an MS-SQL local or NT account that has the privileges as of an administrator). Then the attacker could try and enter other databases and machines and install there more tools that will entrap people and services and allow him access and hiding of its tracks.

How is that the badly written SQL method can get a previously administrator session permissions?

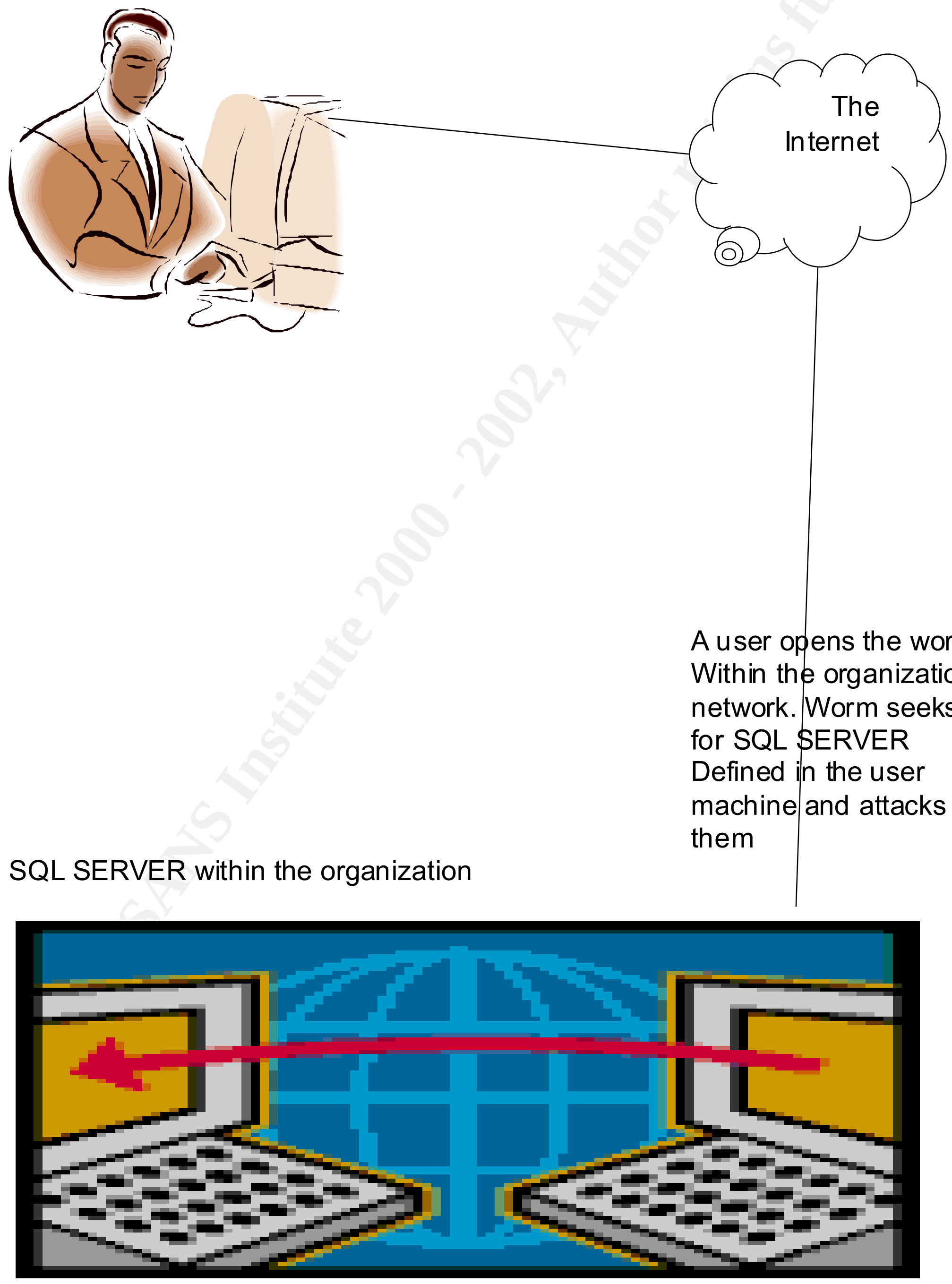
Databases do a lot of activities involving reading and writing information into DISKS (magnetic-storage units that c ould store lots of data). The access to the DISKS is slow in hundreds of percents, compared to doing it at the Computer's memory. So MS -SQL tries to "cache" – that is to put as much data in memory instead of re -read or write it on the DISKS. The MS -SQL database also caches the results many heavy repetitive operations for future usage, including the information of previous login sessions in the Database.
It does it so that if within a relatively short time the Administrator wants to re - login, it is assigned the work space defined by the previous cached session. This way MS-SQL avoids the resource consumption of preparing a new workspace.
The vulnerable MS -SQL method, wrongly assigns a previous cached session workspace to a totally other user logon.

Since the exploit for this vulnerability was not published yet and I am the only known person who created the exploit, I am in discussions with Microsoft regarding the proper way to publish it. Therefore I have not included here the actual exploit code (SQL method) .

I tried the exploit both within local database in a local computer, and from a remote machine that attacked the MS -SQL Database server machine.

Diagram of the attack: (a possible attack)

The attacker computer sends via email
Web pages and other me ans,
A worm containing
Malicious code including malicious SQL statements
And a spreading mechanism (via Mail, Web and others)



A user opens the worm
Within the organization
network. Worm seeks
for SQL SERVER
Defined in the user
machine and attacks
them

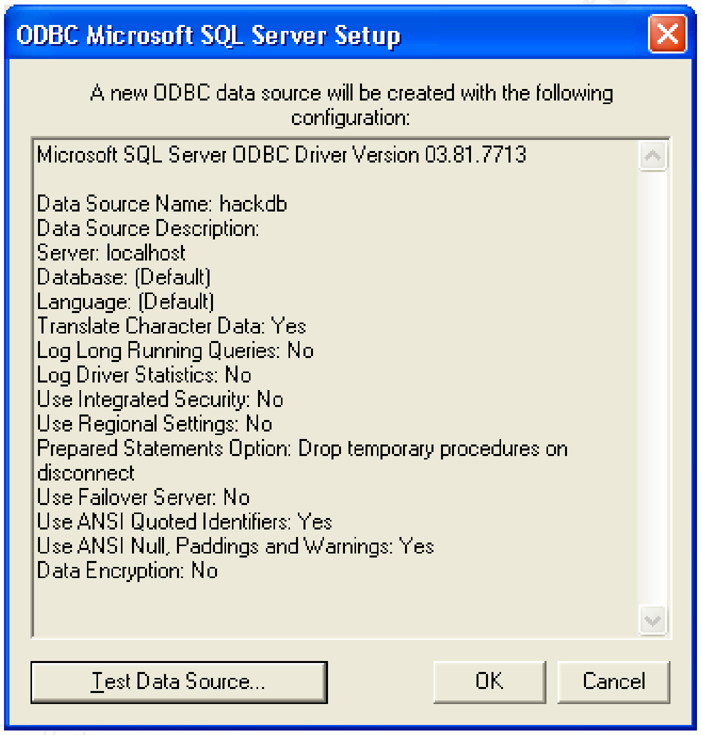
SQL SERVER within the organization

Well worms have proved that they can spread still how will this exploit live and spread within an unknown environment?

There are several ways that allow a hostile code or person to find details of available SQL SERVERS while running on a compromised machine used to connect to those Databases.

1. Interrogate the Machine's repository and disks for well known places that contain definition of connections to Datab ases. Such one is the Control Panel Data Sources definitions that include Database names and Users and Passwords to use on those Databases.
   Also configuration files of applications and log files could be inspected as well for Database connections.

Hereby screen Capture of Data Sources repository of an NT computer: You can notice that the Database server (localhost) and Database name (hackdb) is clearly mentioned.



A new ODBC data source will be created with the following configuration:

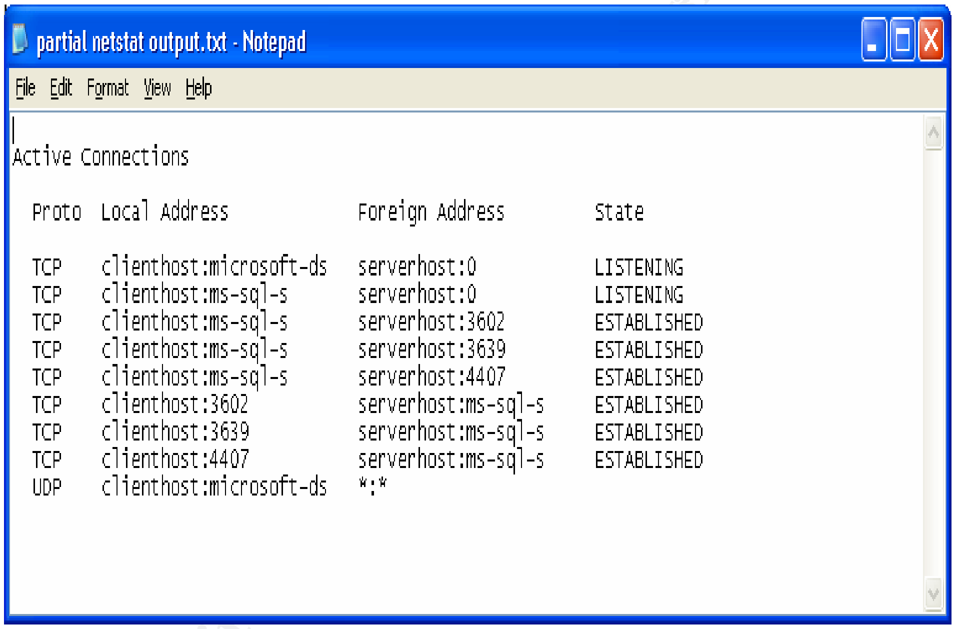
```
Microsoft SQL Server ODBC Driver Version 03.81.7713

Data Source Name: hackdb
Data Source Description:
Server: localhost
Database: (Default)
Language: (Default)
Translate Character Data: Yes
Log Long Running Queries: No
Log Driver Statistics: No
Use Integrated Security: No
Use Regional Settings: No
Prepared Statements Option: Drop temporary procedures on
disconnect
Use Failover Server: No
Use ANSI Quoted Identifiers: Yes
Use ANSI Null, Paddings and Warnings: Yes
Data Encryption: No
```

2. Collect information of current connections that the PC has made on port 1433 (MS-SQL port), using netstat.exe NT command for example.

   Output of Microsoft NT "netstat –a" command shows MS-SQL connections:

   You can see the "Local Address" contains client computer "clienthost" And the TCP or UDP port number or name ("ms -sql-s") or "Microsoft-ds" while the "Foreign Address" column mentions the Server machine name "serverhost" and the port number and/or name that it uses "ms -sql-s".


partial netstat output.txt - Notepad

```
Active Connections

  Proto  Local Address          Foreign Address        State

  TCP    clienthost:microsoft-ds  serverhost:0         LISTENING
  TCP    clienthost:ms-sql-s    serverhost:0           LISTENING
  TCP    clienthost:ms-sql-s    serverhost:3602        ESTABLISHED
  TCP    clienthost:ms-sql-s    serverhost:3639        ESTABLISHED
  TCP    clienthost:ms-sql-s    serverhost:4407        ESTABLISHED
  TCP    clienthost:3602        serverhost:ms-sql-s    ESTABLISHED
  TCP    clienthost:3639        serverhost:ms-sql-s    ESTABLISHED
  TCP    clienthost:4407        serverhost:ms-sql-s    ESTABLISHED
  UDP    clienthost:microsoft-ds  *:*
```

How would one know of a Local MS -SQL database user to use for the exploit?

Well it could find it in while i nterrogating the defined Database connections as mentioned above AND it could try brute force, it could "listen" for port 1433 user connections and sniff the user and password, it could try and use well known user and passwords that are installed as a defa ult while people install commercial products and programs that use MS -SQL as their data repository.

How would one be able to connect to MS -SQL database at all?

One could very easily create a small program that relays upon or includes all necessary items to be able and connect to an MS -SQL Database, not to mention that Computers that use Client programs that connect to MS -SQL Database servers, already have all that is required to access the Database.

What if an organization does not allow computers to dir ectly use MS - SQL?

If the organization users do not directly use an MS -SQL database, they rather use Web Browsers (for example) that send requests to special Web Servers. The Web Servers then connect to MS -SQL Database to execute SQL Code and return result s to the Web Browser. The same mechanism could be exploited to pass the malicious SQL code via the Web Browser to the Web Server, or via break -in initially Web Server using a Web Server Exploit, trying to connect from there to the MS -SQL databases.

Since the exploit for this vulnerability was not published yet and I am the only known person who created the exploit, I am in discussions with Microsoft regarding the proper way to publish it. Therefore I have not included in the following chapters the actua l signature and exact defense for the attack. I will have it published after closing it with Microsoft.

<u>Signature of the Attack:</u>

1. A possible evidence of the attack is the execution of the SQL command that allows the exploit to run. As I explained befor e I will not reveal it here. The SQL code could be identified by an Intrusion Detection system or by automatic inspection of traces for the Database activity, while the MS-SQL Profiler will report of events related to this exploit code. One can observe the malicious SQL using the Process Activity details in the MS -SQL Enterprise Manager utility.
2. Also certain type of Authentication will be carried on by the exploit code that could be identified. I will not reveal the details here. This could be audited by MS-SQL and NT Event log and acted on.

How to protect yourself against the attack and future similar Database Attacks?

1. **Install the appropriate patch** , look at the Microsoft Security bulletin for further details at: <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS01-032.asp>
Installing the patch may require you to first upgrade your MS -SQL level and other related applications that may depend on that.

2. **Consider migrating to Authentication based on NT only** and disable the "Mixed mode" Authentication. You must prepare appropriate users in your NT environment to take the place of the local MS-SQL users and then give those NT-based usernames the proper permissions within the Database. This could be rather not so trivial, however it looks like a very good idea since there are several MS -SQL vulnerabilities that could be avoided as well, if you move from authentications that is "Mixed Mode".

3. **Consider using a Single Sign on system** with strong authentication mechanism that could have stronger overall password maintenance. This may be required, as tying your database as well as other applications to a single source of authorization (Windows-NT authentication) may be less than what your security policy requires. Such a Single Sign On system could hide session login from sniffer malicious programs by encrypting the authentication process.

4. **Verify that your MS-SQL local Database user accounts have a good password policy** and an un-easy to guess password. There are other tools that enable proper account maintenance. Again it seems that you better relay on the NT -Authentication and make sure your NT accounts have an effective security policy (Long, still easy to remember passwords, a proper password -modification timing and tools to provide the users with self-ability to change passwords).

5. **Consider using password cracking tool to test your passwords.** This step must have manage ment approval and consideration for Privacy legal issues.

6. **Operate Auditing for login failure and success** . Having that done, you can configure the NT Application Log Event Viewer or other utilities to prompt you for the Exploit operation or any strange or repeating logins from the same source or user that may reveal an attack trial. Auditing is a good idea for securing and monitoring systems and investigation of events; it requires maintenance and careful consideration to avoid taking too much CPU or DISK r esources for storage and processing of the Audit logs.

7. **Consider using a central AUDIT collection and analysis tool** that gets audits from all systems and stores them remotely such that if a compromise happens on the systems log files, you can view the central system.

8. **The SQL Server has its own set of logs** that contain much more than the Login attempts information and they could be viewed within the MS-SQL Enterprise Manager utility under the Management Section for the SQL server you are checking. They can be also found on DISK.

9. **Run a well configured Intrusion Detection tool in your network and on the MS-SQL servers themselves**. Tools such as public-available "**Snort**" Intrusion Detection system and even a "find string" utility such as "**ngrep**", or commercial tools from companies such as "**eEYE Digital Security**", "**Computer Associates**" and others may allow you to detect this exploit and others in time.

10. **Find your vulnerable systems before they get attacked and fix them**, use public-available vulnerability assessment tools such as "**Nessus**" or commercial tools. Those tools will require your time and effort to configure maintain and fix and they will save you time as well. By avoiding recovery for compromised systems.

11. **Consider using tools that maintain and track Database configuration changes**. Such a tool could alert you in time of an un-authorized modification, possibly carried on by an attacker.

12. **Do regular backups** of your computer files, configuration, Database configuration and files.

13. **Consider operating Database Snapshot Mechanisms** that allow you to take the whole or part of the database "back in time", this consumes resources, still may prove beneficial in recovery and investigation of Database corruption.

14. **Acquire a utility that allows reviewing Database log files** – this will allow you to pinpoint malicious operations and reverse them back.

15. **Consider off-loading the non-active of the database into read-only area** – this will make some of the data be protected from any change including malicious activity – it will be possible for Data that should not be modified by any business process anyways.

16. **Consider operating trace for Database activities** (using the MS-SQL xp_sqltrace command) and use the MS -SQL Server profiler utility to trigger events and alert for the e xploit code (when published) as for other exploits. **Remember that an attack is something to be investigated even if you have the proper patches. That malicious activity should be exterminated before it succeeds** in finding a hole in your "defense wall"! Hav ing the Database trace activity may cause overload on the Database system and should be considered carefully.

17. **Consider avoiding usage of well known accounts** – such as "sa", "administrator" and Applications famous default accounts that they create on your system. Those accounts may be attacked easier to gain un-authorized access. The well known accounts could be sniffed by network sniffing utilities to report of a user and password, while regular accounts may have less attention by intruders.

18. **Inspect your system after each major change and specially after any installation of a product** for possible new holes such as new vulnerable accounts or other database objects.

19. **Consider putting an Access Control system that** tracks sessions and their privileges and that deny access from in-appropriate Client machines that should have no reason to run high -privileged session in the Database system. This may catch the machine that got higher rights in the database. This system may be configured to limit the amount of High-Privileged login sessions to a database, possibly leaving room for the real Administrators only. Also the time of login and other similar parameters could be monitored to prevent access which is absolutely not according to the user normal activities. Comme rcial products such as "Computer Associates eTrust Access Control" may be considered.

20. You could consider denying "Ad Hoc Queries", still my exploit may get around this in specific certain conditions.

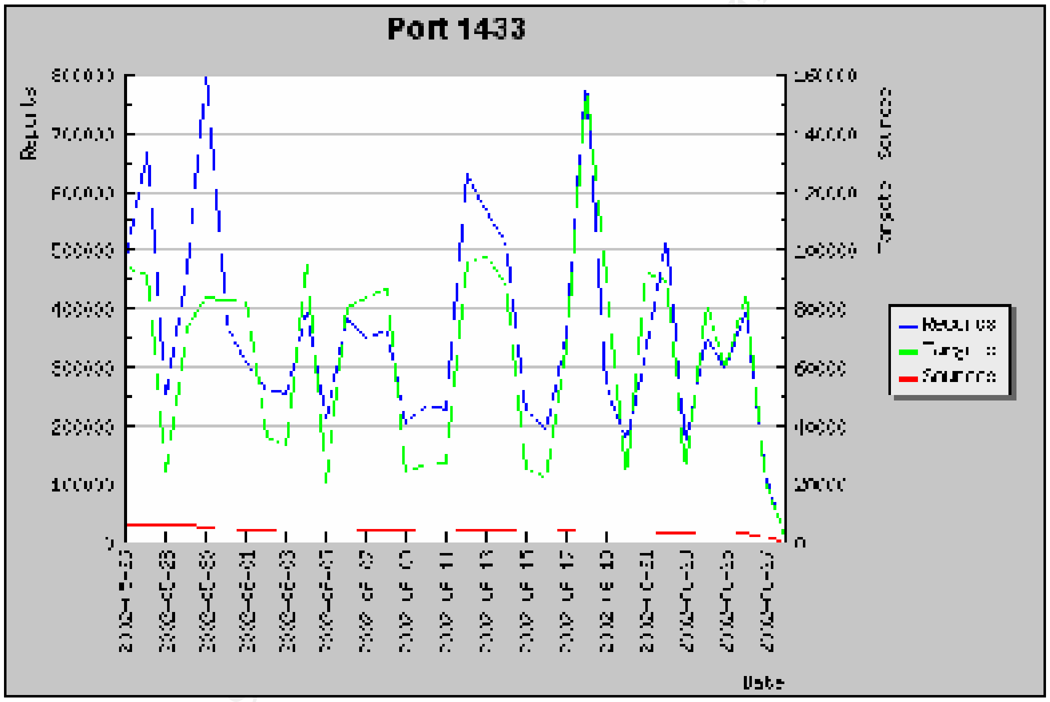
References:

1. The web site http://www.incidents.org – graphs in appendix A were copied from this web site. Also another URL of it at: < http://isc.incidents.org/port_details.html?port=1433 >
2. Microsoft home page – manufacturer of web server IIS (http://www.microsoft.com)
3. Apache Software Foundation home page (http://www.apache.org).
4. Microsoft Security web site (http://www.microsoft.com/security) Oracle
5. Oracle corporation Security Alerts web site (http://otn.oracle.com/deploy/security/alerts.htm)
6. Description of TCP/IP protocol from http://www.protocols.com/pbook/tcpip.htm
7. The Common vulnerabilities and exposures site: http://cve.mitre.org
8. The Computer Incident Advisory Capability site: http://www.ciac.org/ciac/
9. The Microsoft "SQL Server 7.0 Books Online" – review of MS-SQL Services
10. Windows Sockets Network Programming by Bob Quinn and Dave Shute, with foreword by Martin Hall at http://www.sockets.com/toc.htm
11. Named Pipes at Microsoft TechNet: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/rpc/ov-pipes_1pv7.asp>
12. Information about SQL terms and examples at W3schools web site owned by "Refsnes Data" company: <http://www.w3schools.com/sql/sql_intro.asp >
13. The Microsoft vulnerabilities site is located at:<http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin>
14. An article about "SQL Piggyback" written as part of Incident Handler certification, by Matt Borland that can be found at <http://www.giac.org/practical/Matt_Borland_GCIH.zip >
15. Vulnerabilities information from List copied from SecurityFocus vulnerabilities information site at: <http://online.securityfocus.com/bid/2863/info/ >

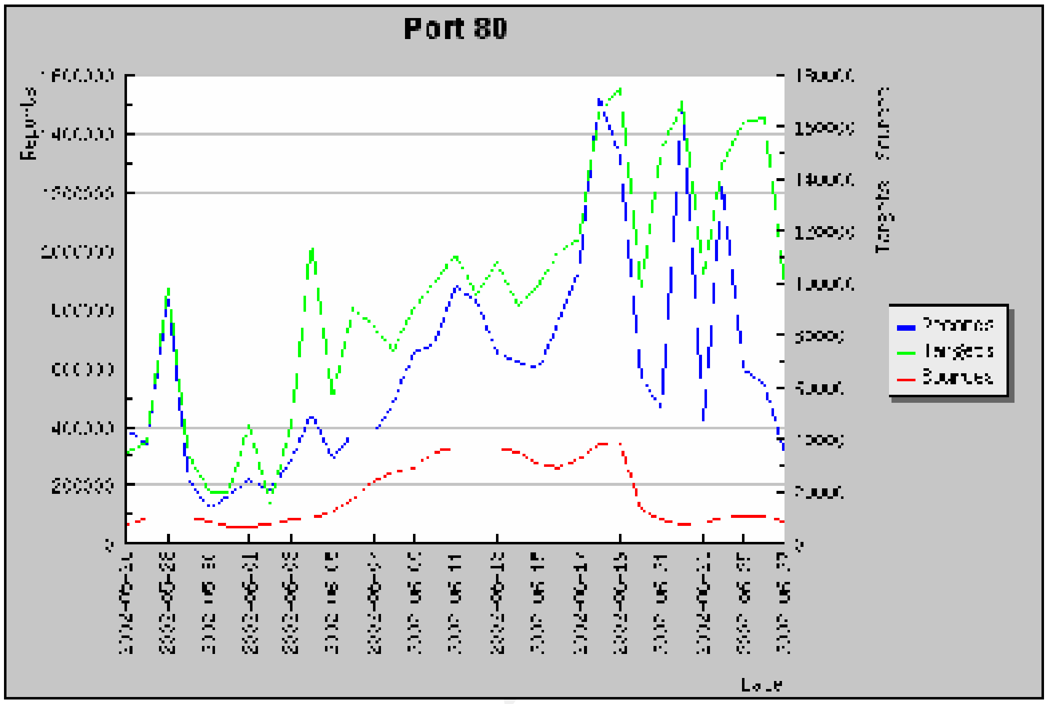
Appendixes:

A. Graphs from the www.incidents.org site, showing statistics regarding amount of attacks on Microsoft SQL port 1433 and other services as comparison.

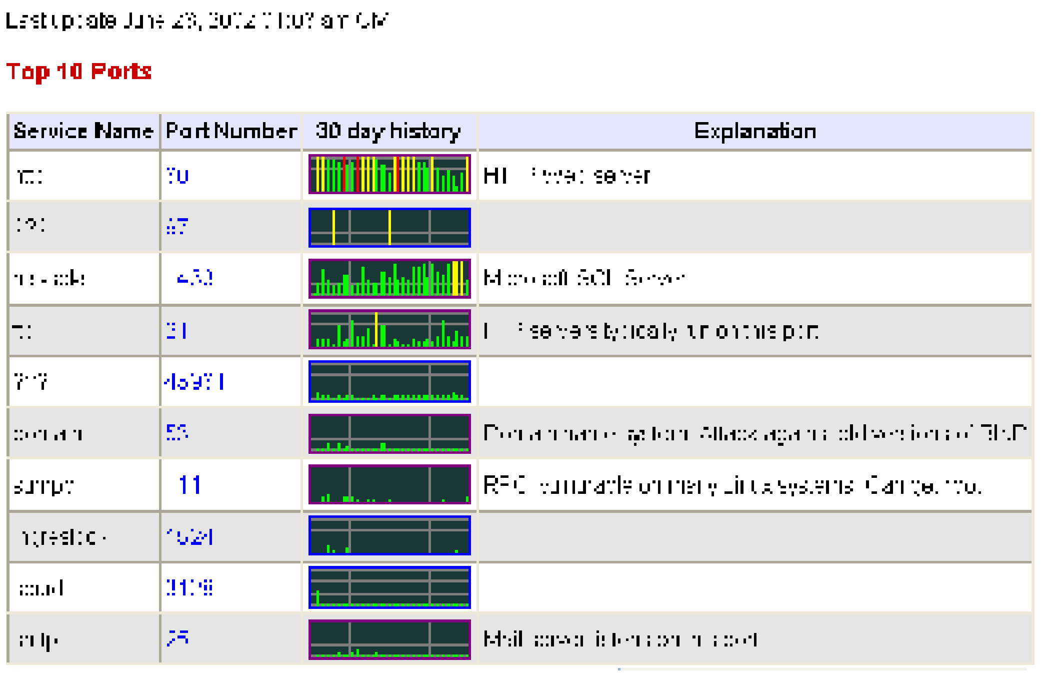
   a. Attacks on Microsoft SQL Database server during the period of 28 - Jun-2002 and 30 days back. There are only about 1 1 known vulnerabilities that are listed for port 1433.



b. Attacks on Web Servers (mostly Microsoft and Apache) during the Period of 28-Jun-2002 and 30 days back. There are at least 150 known vulnerabilities that are listed for port 80.



c. The Top 10 ports attacks during the period of 28 -Jun-2002 and 30 days back. Only port 80 http and port 1433 the most massive and similar attack graph, continuously during all this period



Last update June 28, 2002 11:07 am GMT

Top 10 Ports

Service Name	Port Number	30 day history	Explanation
www	80		HTTP Web server
???	57		
ms-sql-s	1433		Microsoft SQL Server
ftp	21		FTP servers typically run on this port
???	45971		
domain	53		Domain name system. Attacks against old versions of BIND
sunrpc	111		RPC. vulnerable on many Linux systems. Can get root
ingreslock	1524		
squid	3128		
smtp	25		Mail server listens on this port