Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"
at http://www.giac.org/registration/gsec

SQLSnake, Are All your Bases Covered?

GIAC GSEC Practical Ver. 1.4b Option 1
Lenny Dexter
January 15, 2003

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Abstract

In May of 2002 Microsoft SQL Server administrators were advised to make certain that the 'sa' (system administrator) password was properly set in order to prevent an attack that was starting to gain momentum, a worm called SQLSnake. Many diligent administrators confirmed that their servers were properly configured and often didn't give it much more thought. The often-overlooked problem is that we could possibly be running other popular applications that may contain an embedded version of SQL or MSDE with a blank or null 'sa' password and not even be aware of it. The key is becoming aware and identifying vulnerable applications and/or tools that may need to be patched or otherwise protected. Understanding the worm and how it operates, proper testing, patching and/or recovering from an attack along with implementing best SQL security management practices will enable an administrator to take the proper precautions to help prevent this type of attack and others like it.

Background

On both May 20th and 21st of 2002 a sudden spike in the number of TCP/1433 port scans was detected around the world. SQLSnake, also known as SQLSpida.A, SQLSpida.B, Voyager Alpha Force, Hacktool.IPStealer, DoubleTap, Digispud and others was gaining momentum attacking Microsoft SQL Servers. (1) The worm searches for null or blank 'sa' passwords and then begins working it's magic. Microsoft SQL Server version 7.0 and 2000 installations by default do not set the 'sa' account password. To make things worse, Microsoft SQL Server is routinely run with system level permissions. The worm scans networks for machines that are running SQL Server on the default port of 1433 and attempts to make a connection. This is where SQLSnake goes to work on the target system. If successful, it begins by using 'xp_cmdshell' to enable the Guest account and then make it a member of the Local Administrators and Domain Administrators groups. The null 'sa' password is then given a new random password consisting of four alphanumeric pairs with the letters, always in lower case. (2) It's worth noting here that there is a variant (SQLSpida.A) of this worm that changes the 'sqlagentcmdexec' password to the same as the new 'sa' password and then the user is added to both the Local Administrators and Domain Admins Groups on the infected system. It also configures itself to run at system startup. Otherwise it performs the same basic steps outlined above and below. After the account information is modified the worm uses the Guest account to make a connection to the remote drive. A group of helper files (3) are copied to the %SystemRoot% directory of the infected host and properties for these files are set to hidden, these files include:

- system32\drivers\services.exe
- system32\exec.js
- system32\sqlinstall.bat
- system32\sqldir.js
- system32\sqlprocess.js
- system32\run.js

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- system32\timer.dll
- system32\samdump.dll
- system32\clemail.exe
- system32\pwdump2.exe
- system32\sqlexe.js
- system32\sqlexec.exe

Once the files are copied onto the infected host, the modified Guest account is then removed from the Administrators and Domain Admins groups and is set to 'disabled'. The worm now starts another sequence of actions in order to complete the attack. A Windows file named 'netdde.exe' (Network Dynamic Data Exchange) is executed and a connection is made to the hosts SQL Server using the 'sa' account and the newly created password. The 'ipconfig /all' command is executed sending it's output to a text file named 'send.txt'. The 'send.txt file is a collection point for some of the machine information the worm gathers. A listing of database tables is appended to the 'send.txt' file and system password hashes are also copied to the file using the 'pwdump2.exe' utility. www.polivec.com/pwdump3.html.
Next an E-mail is created with a command line mailer program 'Clemail.exe' with the 'send.txt' file attached and sent to a domain registered in Singapore, ixltd@postone.com. The subject line of the E-mail is 'SystemData-[password]' [password] being the newly created 'sa' password.

The worm now enters the final phase of the attack by scanning for other vulnerable hosts. One of the files copied to the %SystemRoot%\system32 directory is 'services.exe' which is Foundstone's 'fscan' www.foundstone.com/knowledge/scanning.html. (Foundstone has replaced 'fscan' with 'scanline' ver. 1.0.1). Fscan is used and the results are saved to a file named 'rdata.exe' which is also located in the %SystemRoot%\system32 directory. The worm examines the result set 'rdata.exe' and IP addresses that are found to have port 1433 open are then sent a copy of the exploit. Finally the 'send.txt' and 'rdata.exe' files are deleted from the host.

Besides the obvious server data and configuration information that was compromised during the attack, the worm can cause a Denial-Of-Service condition on the network while the worm is scanning for additional hosts to attack. The 'fscan' scanner scans both internal and external IP addresses. Fscan is set to use100 threads creating a lot of extra network traffic. The attacker could also use this vulnerability to run arbitrary commands on the compromised host or launch additional attacks on other systems masquerading as the infected host. This worm is somewhat more advanced when compared to previous SQL worms.

So What's The Problem?

To the average busy administrator this may sound like the fix is fairly simple, just make sure your Microsoft SQL Servers have strong 'sa' passwords set and that's it. There are, of course, other important steps that need to be taken to properly protect against this type of vulnerability. This paper will discuss this in more detail

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

later on. The problem is this vulnerability exists in other commonly used applications where some administrators wouldn't think twice about looking. The fix used to patch Microsoft SQL Servers (setting a strong 'sa' password) is not an option for some of the other applications vulnerable to this attack. Some application vendors elected to issue warnings and patches while it seems others have yet to do so. The SQL MSDE (Microsoft SQL Server Desktop Engine) is not only used in Microsoft products but a runtime version is licensed and embedded by software vendors in their applications as well. MSDE gets installed as a component of some applications without the user ever being aware of it. Some of the more popular applications known to use MSDE are:

- Microsoft Visio 2000 Enterprise
- Microsoft Visio Enterprise Network Tools
- Microsoft Office 2000/XP on Windows 98/ME
- Microsoft Project Central
- Microsoft Visual Studio
- Microsoft Access 2000
- Dell IT Assistant Ver. 6.2
- Compaq Insight Mgr. 7 (all ver.)
- Compaq Insight Mgr. XE (1.1+)
- IBM Director 3.1

Risk Evaluation and Testing.

There are some methods to test and see if a suspect 'MSDE embedded' application is at risk. Most of the vendors that incorporate the Microsoft SQL MSDE product have now acknowledged the problem and have provided a solution or patch while other vendors have not made it as easy. If you suspect that you may have a product that meets the above criteria, you should be able to use the following tests to check for the vulnerability. Start by port scanning your network for any machines that may be listening on TCP port 1433 to determine which hosts are vulnerable and need further testing. There are free tools available to aid administrators in finding vulnerable machines and most provide some advice on repairing hosts that have been infected.

Vulnerability Scanners

A free SQLSnake scanner is available from Eeye Digital Security located at: www.eeye.com/html/Research/Tools/Retinasqlworm.exe. This tool called 'RetinaSqlWorm' is a 640k download that will scan up to 254 IP addresses at one time looking for the blank 'sa' password vulnerability. It is a simple GUI interface that allows the user to enter IP address ranges and view the results designated as either 'infected' or 'not infected'. This tool does not repair an infected machine but provides links to a website that walks administrators step by step through removing the worm. RetinaSqlWorm is part of a larger full-featured network security scanner product.

Another well thought out free scanner is PentaSafe's SQL Snake Bite KitA quick non-intrusive registration and software license agreement is all that is required to obtain this tool. This scanner is different from Retinasqlworm.exe in that it features an Auto Discovery tool, which like the name suggests finds and identifies

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

vulnerable hosts automatically. It works the same as the scanner from Eeye.com, by attempting to login to hosts as 'sa' with a null password.

Microsoft's Solution

Microsoft provides some solutions for this vulnerability. It is interesting that Microsoft's position regarding this vulnerability is that the blank 'sa' password is not a bug in their software, but rather an end user configuration issue. I performed a number of lab installations of SQL Server while researching this paper and found that I was prompted to enter an 'sa' password but was allowed to not enter a password as well, effectively leaving it blank. The following procedures will allow an administrator to patch Microsoft products that contain the Microsoft SQL Server Desktop Engine (MSDE 1.0) functionality.

Verify that your 'sa' password is in fact blank.

Start a command prompt and enter the following command:
osql –U sa (enter) or use the following command to connect to a specific MSDE install name:
osql – U sa –S servername\specific_MSDE_name.
The system will return a 'Password' prompt. Hit the Enter key that will attempt to set the 'sa' password to 'NULL'.
If you are presented with a '1>' prompt then your password is in fact blank and you will need to set it. If instead you receive the following error message: 'Login Failed for user 'sa', then you have entered an invalid password which really means that an 'sa' password has already been set.
If you receive: 'Login Failed for user 'sa', Reason: Not associated with a trusted SQL Server Connection', then you will be unable to verify the 'sa' password because your product is running in Windows Authentication Only mode instead of Mixed Mode.
Note; Microsoft recommends that administrators not use Mixed Mode Authentication if possible and to rely on Windows Authentication because of the added security benefits.
You will still be able to create an 'sa' account password to make sure you are not vulnerable in case your authentication scheme is converted to Mixed Mode in the future.
If you received the following results then it is possible that the application may not be running or you have provided an incorrect name for the named instance of SQL server that is installed:

[Shared Memory] SQL Server does not exist or access is denied.
[Shared memory] ConnectionOpen (Connet()).

Setting A Proper 'sa' Password.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

If the testing performed in the last section determined that the 'sa' password is blank, follow the next few steps to set a proper password. It's said that a strong password is eight or more characters in length using a combination of both upper and lower case letters, numbers and characters.

Start a command prompt and enter the following command:
osql –U sa (or osql –E if you are using Windows authentication.) (enter)
Enter the following on two separate lines and press Enter:
sp_password @old = null, @new = 'new_proper_passwd', @loginame = 'sa'
go
Result message will be: Password Changed

Visio Enterprise Edition and Enterprise Network Tools

Some Microsoft Visio products and legacy Visio products are installed with MSDE version 1.0. Also having a blank 'sa' password, these include Visio 2000 Enterprise Edition and Microsoft Visio 2000 Enterprise Edition SR-1. Also affected is Microsoft Visio Enterprise Tools (also known as VENT) with MSDE 2000 installed. The testing and patching procedures are the same as the previous section for Microsoft products. (4)

Compaq Insight Manager 7.0 & XE

Compaq's Insight Manager products did not install MSDE by default but it was available on the distribution CD for users that needed the MSDE functionality. Users can manually install MSDE, and that is where the problem begins. The 'sa' password is left blank and without manual intervention, the vulnerability exists. The testing and patching procedures are different from the Microsoft solution above.

1. Login to Insight Manager go to Settings\Server\Database and change the password to your new strong password.
2. Stop the Compaq Insight Manager Service and open a command prompt.
3. In the MSSQL7 directory type 'osql –Usa –P (return) (logs in with null password.
4. Type sp_password @old="", @new='strong_password', @loginame = sa (enter).
5. Type go (return) and a 'Password changed' message will appear.
6. Restart Insight Manager 7 service.
7. If the password was not changed, Insight Manager will fail to start. At this point Compaq advises that users contact Compaq support. (5)

Dell IT Assistant Ver. 6.2

1. Make sure Dell IT Assistant is running. At a command prompt type the following:

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Osql –U sa –P (enter) (the prompt should change to '1>')

2. Enter the following command:

   1> EXEC sp_password NULL, 'password' , 'sa' (enter)
   2> go (return)

After hitting return you should get the result 'password changed'. If after hitting return in step one you do not receive the '1>' prompt and instead get a password error message, that is an indication that the application has an 'sa' account password already set. If you receive the errors below then it is possible that the application may not be running or you have provided an incorrect name for the named instance of SQL server that is installed. [Shared Memory] SQL Server does not exist or access is denied or [Shared memory] ConnectionOpen (Connect()). (6)

Recovering From SQLSnake

The first step is to confirm that you have in fact been infected. There are a number of ways to detect this. The worm copies a group of helper files (see file listing in the beginning of this paper) to the %SystemRoot%\system32 directory. A quick search for these files would be all that is needed to know if a host was in fact a victim. Because the worm sets the helper files to 'hidden', you will need to use 'attrib' from a command prompt in the system directory to expose them. To remove the worm and restore the machine, the first thing to do is change the 'sa' password. Next confirm that the Guest account is disabled. Because part of the worm's return payload is the host's password hashes, all accounts need to have their passwords changed. The worm makes some modifications to the registry and the following keys will need to be removed:

HKEY_LOCAL_MACHINES\System\CurrentControlSet\Services\NetDDE\ImagePath
HKEY_LOCAL_MACHINES\System\CurrentControlSet\Services\NetDDE\Start
HKEY_LOCAL_MACHINES\software\microsoft\mssqlserver\client\connectto\dsquery

The helper files need to be deleted. Using the file listing in the beginning of this paper, run the 'attrib –h' command to remove the files hidden property. Now that the files are exposed, simply delete them. Lastly, the worm also registers the helper file called 'timer.dll'. Unregister this file using the following command: regsvr32 /u TIMER.DLL

The easier method is to use a specific tool to remove the worm and help restore a host. There's a simple but effective free utility called 'SQLSnake Removal Tool'. Written by Felipe Moniz, this tool scans the local host for the worm's helper files and if found removes the files and reverses any of the worm's modifications to the registry. This tool can be freely downloaded from www.webattack.com/get/sqlsnakeremover.shtml. (7)

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Best Security Practices for SQL Server Installations.

Installing service packs and hot fixes. Test service packs and hot fixes in a lab environment to check for compatibility issues before installing on production systems. Since this paper is primarily focused on the vulnerabilities relating to the blank 'sa' password, I will start by saying this is one of the easiest and quickest items to make secure. The procedures outlined earlier will help administrators to make sure that a strong 'sa' password is in place. Even those administrators who never login with the 'sa' account will need to check this. The Microsoft SQL Server Books Online is generally installed when the server software is installed and makes a great reference for administrators wishing to better understand the System Administrator account.

Blocking port 1433 at the gateway. To prevent port scans that can identify SQL products listening on the default port. It stands to reason that when an attacker is port scanning for a specific vulnerability and doesn't get a hit he or she will move on to the next network. Port scan your network from the outside for port 1433 to identify any machines that may be vulnerable and to confirm that the port is effectively being blocked.

Block outbound E-mail destined for ixltd@postone.com, which is SQLSnake's result payload destination.

Changing the default port. If an administrator has to run an SQL Server facing the Internet, changing the default port the server listens on to another port can significantly help to prevent worms like SQLSnake from being as effective. If the SQL server must use port 1433 and be available on the Internet, performing egress filtering at the firewall can be very effective at monitoring and preventing outbound port 1433 connections to other systems, which would otherwise continue to spread the worm.

While researching this paper I took this technique into a lab network in order to test our corporate SQL based application. The application was listening on the default port. I modified the SQL server settings to start using a new port other than the default. I configured the gateway router and firewall to allow the newly chosen port number through to this specific host's address. I then configured logging on the firewall to log both successful and unsuccessful connection attempts. I also set up egress filtering to add an additional layer of security. Modifications were made to the ODBC connection string on the client side. This allows an administrator to change the port that the client application uses when making a connection to the server. I then blocked port 1433 at the firewall but continued to log port scans for port 1433 at the firewall to do a comparison with logs from when 1433 wasn't being blocked. The first thing I noticed was that the pattern of multiple probe attempts from the same source address was dramatically reduced. It appears that since port 1433 wasn't answering the

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

probes, the attackers moved on instead of making additional attempts at known vulnerabilities. The logs on the SQL Server showed that there were no failed login attempts from unknown source addresses, which was a huge improvement, compared to archived logs of when the server was listening on port 1433. After a successful period of testing an enterprise wide change was made to all of our corporate SQL servers, clients and related networking hardware in order to deploy the new port number. There was a lot of planning and work involved to roll out these changes, but considered well worth it after comparing firewall and server logs. I determined that we greatly reduced the chance of being identified as target for running Microsoft SQL Server vulnerabilities against.

Running both the SQLServer service and SQL Server Agent under a Microsoft Windows account instead of a Local System Account gives an added level of protection.

If possible, enable Microsoft Windows Authentication as well as enabling auditing looking for both unsuccessful and successful logins. The MSSQLServer service will need to be stopped and restarted in order for the changes to take affect. You will want to modify your clients to start using Windows Authentication as well.

It is worth noting here that when Microsoft SQL Server 2000 is installed, the installation prompts the user to choose between SQL Server / Windows Authentication known as 'Mixed Mode', or 'Windows Authentication Only'. If the user chooses Mixed Mode a password prompt allows the user to enter an 'sa' password. If Windows Only mode is chosen during installation, and sometime in the future it is changed to Mixed Mode the 'sa' password will be left blank. Unfortunately Enterprise Manager does not give the user a way to enter an 'sa' password in this situation. [8] Using the stored procedure 'sp_password' in Query Analyzer will allow the administrator to fix this. Running the following syntax will set an 'sa' password:

sp_password NULL, 'NewPassword', 'sa'

This vulnerability was fixed in SQL 2000 SP1. Additional information regarding 'sp_password' can be found in the SQL Server Books Online documentation.

Use strong passwords and change them periodically. Simply confirming that all SQL servers and applications have an 'sa' password is not quite enough; passwords need to have strength in order to help slowdown password cracking tools. A strong password is said to have eight or more characters in length using a combination of both upper and lower case letters, numbers and characters. Since most cracking tools can eventually crack even strong passwords, it's a good idea to periodically change passwords to add an additional layer of defense.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Conclusion

SQLSnake is a perfect example of how even though we diligently install security patches; other applications that are just as susceptible to the same vulnerabilities in our networks can be often overlooked. It's clear that Securing SQL Server requires multiple layers of protection coupled with an ongoing awareness of the latest vulnerabilities. Could Microsoft have forced users to choose a strong password during installation, maybe, but as always security is ultimately the responsibility of the individual of the administrator.

Researching this paper made it quite evident that no single source of information on this topic provides all of the answers. An administrator must dig a little in order to get all of the pieces required to completely understand this and other vulnerabilities. The user community has greatly enhanced the effort of software vendors when it comes to keeping administrators informed.

© SANS Institute 2003, Author retains full rights

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

References

[1] Millard, Elizabeth. "SQL Server Worm: Just the Beginning" May 22, 2002. (December 15, 2002)
URL: http://www.newsfactor.com/perl/story/17893.html

[2] Eeye Digital Security. "SQL Worm Analysis May 22, 2002" (December 17, 2002)
URL: http://www.eeye.com/html/research/Advisories/AL20020522.html

[3] E-Secure-DB. "Update 38 Virus& Security Watch" from IDGNet NZ May 24, 2002 (January 7, 2003)
URL: http://www.e-secure-db.us/dscgi/ds.py/View/Collection-244

[4] Microsoft Knowledge Base Article 321081. "Visio: Installation of MSDE Creates an 'sa' Account with a Blank Password" August 6, 2002 (November 12, 2002)
URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;321081

[5] Neohapsis Archives. "Compaq Insight Manager & Potential SQL Server / MSDE Security Vulnerability". June 2002 (November 12, 2002)
URL: http://archives.neohapsis.com/archives/compaq/2002-q2/0111.html

[6] Noel, Stan. Dell Community Forum. "Blank sa Vulnerability in IT Assistant Running MSDE." May 30, 2002 (December 15, 2002)
URL:
http://forums.us.dell.com/supportforums/board/message?board.id=pes_oms&message.id=1749

[7] WebAttack.com. "SqlSnake Removal Utility". September 9, 2002 (December 15, 2002)
URL: http://webattack.com/get/sqlsnakeremover.shtml

[8] "If You Change Windows Security to Windows / SQL Security the SA Password is Blank". October 15, 2002. Microsoft Knowledge Base Article – 274773. (January 5, 2003)
URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;274773

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.