Overview

There are many ways to attack a database. External attacks may exploit configuration weaknesses that expose the database server. An insecure Web application may also be used to exploit the database. For example, an application that is granted too much privilege in the database or one that does not validate its input can put your database at risk.

Internal threats should not be overlooked. Have you considered the rogue administrator with network access? What about the database user tricked into running malicious code? For that matter, could any malicious code on the network compromise your database?

This chapter begins by reviewing the most common threats that affect database servers. It then uses this perspective to create a methodology. This chapter then puts the methodology into practice and takes a step-by-step approach that shows you how to improve your database server's security.

How to Use This Chapter

This chapter provides a methodology and steps for securing a database server. The methodology can be adapted for your own scenario. The steps put the methodology into practice.

To gain the most from this chapter:

* **Read Chapter 2, "Threats and Countermeasures."** This chapter provides an explanation of potential threats faced by Web applications and downstream database servers.
* **Use the snapshot**. The section, "Snapshot of a Secure Database Server," later in this chapter lists the attributes of a secure database server. It reflects distilled input from a variety of sources including customers, industry experts, and internal Microsoft development and support teams. Use the snapshot table as a reference when configuring your database server.
* **Use the checklist**. The "Checklist: Securing Your Database Server" in the "Checklist" section of this guide provides a quick reference. Use the checklist to quickly evaluate the scope of the required steps and to help you work through the individual steps.
* **Use the "How To" section**. The "How To" section in this guide includes the following instructional articles that help you implement the guidance in this chapter:
  + "How To: Use Microsoft Security Baseline Analyzer"
  + "How To: Use IPSec"
  + "How To: Implement Patch Management"

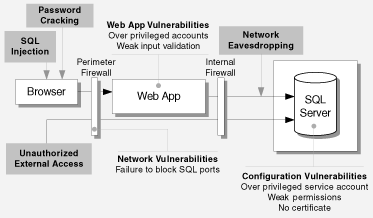
Threats and Countermeasures

An attacker can target and compromise a database server in a number of ways by exploiting a variety of configuration and application level vulnerabilities.

The main threats to a database server are:

* **SQL injection**
* **Network eavesdropping**
* **Unauthorized server access**
* **Password cracking**

Figure 18.1 shows the major threats and vulnerabilities that can result in a compromised database server and the potential destruction or theft of sensitive data.



**Figure 18.1**

***Top database server threats and vulnerabilities***

The next sections describe each of these threats and vulnerabilities.

SQL Injection

With a SQL injection attack, the attacker exploits vulnerabilities in your application's input validation and data access code to run arbitrary commands in the database using the security context of the Web application.

Vulnerabilities

Vulnerabilities exploited by SQL injection include:

* Poor input validation in your Web applications
* Unsafe, dynamically constructed SQL commands
* Over-privileged application logins to the database
* Weak permissions that fail to restrict the application's login to the database

Countermeasures

To counter SQL injection attacks:

* Your application should constrain and sanitize input data before using it in SQL queries.
* Use type safe SQL parameters for data access. These can be used with stored procedures or dynamically constructed SQL command strings. Using SQL parameters ensures that input data is subject to type and length checks and also that injected code is treated as literal data, not as executable statements in the database.
* Use a SQL Server login that has restricted permissions in the database. Ideally, you should grant execute permissions only to selected stored procedures in the database and provide no direct table access.

For more information about application-level countermeasures to SQL injection attacks, see Chapter 14, "[Building Secure Data Access](https://msdn.microsoft.com/en-us/library/aa302430.aspx)."

Network Eavesdropping

The deployment architecture of most applications includes a physical separation of the data access code from the database server. As a result, sensitive data, such as application-specific data or database login credentials, must be protected from network eavesdroppers.

Vulnerabilities

Vulnerabilities that increase the likelihood of network eavesdropping include:

* Insecure communication channels
* Passing credentials in clear text to the database; for example:
  + Using SQL authentication instead of Windows authentication
  + Using SQL authentication without a server certificate

Countermeasures

To counter network eavesdropping:

* Use Windows authentication to connect to the database server to avoid sending credentials over the network.
* Install a server certificate on the database server. This results in the automatic encryption of SQL credentials over the network.
* Use an SSL connection between the Web server and database server to protect sensitive application data. This requires a database server certificate.
* Use an IPSec encrypted channel between Web and database server.

Unauthorized Server Access

Direct access to your database server should be restricted to specific client computers to prevent unauthorized server access.

Vulnerabilities

Vulnerabilities that make your database server susceptible to unauthorized server access include:

* Failure to block the SQL Server port at the perimeter firewall
* Lack of IPSec or TCP/IP filtering policies

Attacks

Direct connection attacks exist for both authenticated users and those without a user name and password; for example:

* Tools such as Query Analyzer (Isqlw.exe) or the command line equivalent (Osql.exe) are used to establish a direct connection to SQL Server and issue commands.
* Server information, such as software version, is revealed to an attacker who sends carefully constructed packets to listening ports.

Countermeasures

To counter these attacks:

* Make sure that SQL Server ports are not visible from outside of the perimeter network.
* Within the perimeter, restrict direct access by unauthorized hosts, for example, by using IPSec or TCP/IP filters.

Password Cracking

A common first line of attack is to try to crack the passwords of well known account names, such as **sa** (the SQL Server administrator account).

Vulnerabilities

Common vulnerabilities that lead to password cracking are:

* Weak or blank passwords
* Passwords that contain everyday words

Attacks

Common password cracking attacks include:

* Dictionary attacks
* Manual password guessing

Countermeasures

To counter these attacks:

* Create passwords for SQL Server login accounts that meet complexity requirements.
* Avoid passwords that contain common words found in the dictionary.

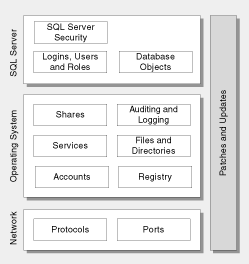
**Note**   If you use Windows authentication, password complexity can be enforced by Windows security policy.

Methodology for Securing Your Server

Securing SQL Server and Windows 2000 or Windows Server 2003 involves many configuration changes. The best approach is to separate the changes that must be made into specific configuration categories. Using categories allows you to systematically walk through the securing process from top to bottom or pick a particular category and apply specific steps.

Configuration Categories

The securing methodology has been organized into the categories shown in Figure 18.2.



**Figure 18.2**

***Database server security categories***

The configuration categories shown in Figure 18.2 are based on best practices obtained from field experience, customer validation, and the study of secure deployments. The rationale behind the categories is as follows:

* **Patches and Updates**

Many security threats exist because of vulnerabilities in operating systems, services, and applications that are widely published and well known. When new vulnerabilities are discovered, attack code is frequently posted on Internet bulletin boards within hours of the first successful attack. Patching and updating your server's software is the first step toward securing your database server. There may be cases where a vulnerability exists and no patch is available. In these cases, be aware of the details of the vulnerability to assess the risk of attack and take measures accordingly.

* **Services**

Services are prime vulnerability points for attackers who can exploit the privileges and capabilities of the service to access the server and potentially other computers. Some services are designed to run with privileged accounts. If these services are compromised, the attacker can perform privileged operations. By default, database servers generally do not need all services enabled. By disabling unnecessary and unused services, you quickly and easily reduce the attack surface area.

* **Protocols**

Limit the range of protocols that client computers can use to connect to the database server and make sure you can secure those protocols.

* **Accounts**

Restrict the number of Windows accounts accessible from the database server to the necessary set of service and user accounts. Use least privileged accounts with strong passwords in all cases. A least privileged account used to run SQL Server limits the capabilities of an attacker who compromises SQL Server and manages to execute operating system commands.

* **Files and Directories**

Use NTFS file system permissions to protect program, database, and log files from unauthorized access. When you use access control lists (ACLs) in conjunction with Windows auditing, you can detect when suspicious or unauthorized activity occurs.

* **Shares**

Remove all unnecessary file shares, including the default administration shares if they are not required. Secure any remaining shares with restricted NTFS permissions. Although shares may not be directly exposed to the Internet, a defense in depth strategy with limited and secured shares reduces risk if a server is compromised.

* **Ports**

Unused ports are closed at the firewall, but it is required that servers behind the firewall also block or restrict ports based on their usage. For a dedicated SQL Server, block all ports except for the necessary SQL Server port and the ports required for authentication.

* **Registry**

SQL Server maintains a number of security-related settings, including the configured authentication mode in the registry. Restrict and control access to the registry to prevent the unauthorized update of configuration settings, for example, to loosen security on the database server.

* **Auditing and Logging**

Auditing is a vital aid in identifying intruders, attacks in progress, and to diagnose attack footprints. Configure a minimum level of auditing for the database server using a combination of Windows and SQL Server auditing features.

* **SQL Server Security**

A number of SQL Server security settings can be controlled through Enterprise Manager. These include the authentication mode, auditing level, and the accounts used to run the SQL Server service. For improved security, you should use Windows authentication. You should also enable SQL Server logon auditing and ensure that the SQL Server service runs using a least privileged account.

* **SQL Server Logins, Users, and Roles**

SQL Server 2000 manages access control using logins, databases, users, and roles. Users (and applications) are granted access to SQL Server by way of a SQL server login. The login is associated with a database user and the database user is placed in one or more roles. The permissions granted to the role determine the tables the login can access and the types of operations the login can perform. This approach is used to create least privileged database accounts that have the minimum set of permissions necessary to allow them to perform their legitimate functionality.

* **SQL Server Database Objects**

The ability to access SQL Server database objects, such as built-in stored procedures, extended stored procedures and **cmdExec**jobs, should be reviewed. Also, any sample databases should be deleted.

SQL Server Installation Considerations

Before taking steps to secure your database server, know the additional components that are present on a Windows 2000 or Windows Server 2003 server after SQL Server is installed.

What Does SQL Server Install?

When you install SQL Server, a number of Windows services are installed in addition to program and data files. By default, program and data files are located in the \Program Files\Microsoft SQL Server\ directory. Table 18.1 shows the services and folders that are created.

**Table 18.1   SQL Server Installation Defaults**

|  |  |
| --- | --- |
| **Item** | **Details** |
| Services | MSSQLSERVER MSSQLServerADHelper Microsoft Search SQLSERVERAGENT |
| Folders | \program files\Microsoft SQL Server\mssql\binn (program files) \program files\Microsoft SQL Server\mssql\data (data files including .mdf, .log, and .ndf) \program files\Microsoft SQL Server\80\tools (shared tools/books online) \program files\Microsoft SQL Server\mssql\logs (error logs) \program files\Microsoft SQL Server\mssql\backup (backup files) \program files\Microsoft SQL Server\mssql\jobs (temp job output files)  For named instances, the instance name is used in the file path:  \program files\Microsoft SQL Server\MSSQL$InstanceName\binn \program files\Microsoft SQL Server\MSSQL$InstanceName\data |

SQL Server Installation Recommendations

If you are building a new database server from scratch, there are a number of considerations to take into account before installing SQL Server. Also, it is a good idea to perform a custom installation of SQL Server so you can select the most secure installation options.

Before Running SQL Server Setup

Before you run the SQL Server setup program, check the following items:

* Create a least privileged local account with which to run the SQL Server service. Use this account when you are prompted for service settings during setup. Do not use the local system account or an administrator account.
* Make sure you do not install SQL Server on a domain controller.
* Make sure you install SQL Server on a partition formatted with NTFS.
* Install SQL Server program and database files on a non-system volume, separate from the operating system.

Installing SQL Server

When installing SQL Server on a production server, choose the custom setup option. When you do this, you can selectively choose the items to install. You should not install the items listed in Table 18.2 on a production database server.

**Table 18.2   Items Not to Install During Custom Installation**

|  |  |
| --- | --- |
| **Tool** | **Purpose** |
| Upgrade tools | Used to upgrade SQL Server 6.5 databases |
| Replication support | Script and binary files used for replication. (Do not install unless you need replication.) |
| Full text search | Full text search engine (Microsoft Search service). Do not install unless you require full text search. |
| Books online | SQL Server documentation |
| Development tools | Headers and library files used by C developers and Microsoft Data Access (MDAC), and XML software development kits (SDKs), and an interface for stored procedure debugging. |
| Code samples | Sample code used to educate developers. |

Also, select Windows authentication mode unless SQL Server authentication is specifically required. Windows authentication offers the following advantages:

* Existing domain and local security policies can be used to enforce strong passwords and account management best practices.
* Credentials are not passed over the network.
* Application database connection strings do not require credentials.

If you select Mixed Mode, create a strong password for the sa account. The sa account is a prime target for password guessing and dictionary attacks.

Steps for Securing Your Database Server

This section guides you through the process of securing your database server using the configuration categories introduced earlier. The steps cover Windows 2000 and Windows Server 2003 and SQL Server 2000. Each step may contain one or more actions to secure a particular area or feature.

|  |  |  |  |
| --- | --- | --- | --- |
| **Step 1**  **Step 2**  **Step 3**  **Step 4**  **Step 5**  **Step 6** | Patches and Updates  Services  Protocols  Accounts  Files and Directories  Shares | **Step 7**  **Step 8**  **Step 9**  **Step 10**  **Step 11**  **Step 12** | Ports  Registry  Auditing and Logging  SQL Server Security  SQL Server Logins, Users, and Roles  SQL Server Database Objects |

Step 1. Patches and Updates

Failure to apply the latest patches and updates in a timely manner means that you are providing opportunities for attackers to exploit known vulnerabilities. You should verify that your database server is updated with the latest Windows 2000 / Windows Server 2003 and SQL Server service packs and updates.

**Important**   Make sure to test patches and updates on test systems that mirror your production servers as closely as possible before applying them on production servers.

Detect Missing Service Packs and Updates

Use the Microsoft Baseline Security Analyzer (MBSA) to detect the necessary Windows and SQL Server updates that may be missing. MBSA uses an XML file as the reference of existing updates. This XML file is either downloaded by MBSA when a scan runs, or the file can be downloaded on the local server or from a network server.

**To detect and install patches and updates**

1. Download and install MBSA.

You can do this from the MBSA home page at [http://technet.microsoft.com/en-us/security/cc184924.aspx](https://technet.microsoft.com/en-us/security/cc184924.aspx).

If you do not have Internet access when you run MBSA, it will not be able to retrieve the XML file containing the latest security settings from Microsoft. In this event, download the XML file manually and put it in the MBSA program directory. The XML file is available from [http://technet.microsoft.com/en-us/security/cc184923.aspx](https://technet.microsoft.com/en-us/security/cc184923.aspx).

1. Run MBSA by double-clicking the desktop icon or selecting it from the **Programs** menu.
2. Click **Scan a computer**. MBSA defaults to the local computer.
3. Clear all check boxes apart from **Check for security updates**. This option detects which patches and updates are missing.
4. Click **Start scan**. Your server is now analyzed. When the scan is complete, MBSA displays a security report, which it also writes to the %userprofile%\SecurityScans directory.
5. Download and install the missing updates.

Click the **Result details** link next to each failed check to view the list of security updates that are missing. The resulting dialog box displays the Microsoft security bulletin reference number. Click the reference to find out more about the bulletin and to download the update.

For more information about using MBSA, see "[How To: Use the Microsoft Baseline Security Analyzer](https://msdn.microsoft.com/en-us/library/aa302360.aspx)" in the "How To" section of this guide.

For more information about applying service packs, hot fixes, and security patches, see [http://technet.microsoft.com/en-us/library/cc750077.aspx](https://technet.microsoft.com/en-us/library/cc750077.aspx).

Patching MSDE

The Microsoft Desktop Edition (MSDE) of SQL Server must be patched differently than the full version of SQL Server. For details about patching MSDE, see "[Secure Your Developer Workstation](https://msdn.microsoft.com/en-us/library/aa302367.aspx)" in the "How To" section of this guide.

Step 2. Services

To reduce the attack surface area and to make sure you are not affected by undiscovered service vulnerabilities, disable any service that is not required. Run those services that remain using least privileged accounts.

In this step, you:

* **Disable unused SQL Server services**.
* **Disable the Microsoft DTC (if not required)**.

**Note**   To disable a service, set its startup type to **Disabled** using the Services MMC snap-in in the Computer Management tool.

Disable Unused SQL Server Services

During a SQL Service installation, the following four Windows services are installed:

* **MSSQLSERVER** (or MSSQL$InstanceName for a named instance). This is the SQL Server database engine and is the only mandatory service.
* **SQLSERVERAGENT** (or SQLAgent$InstanceName for a named instance). With this support service, you can schedule commands and notify operators when errors occur.
* **MSSQLServerADHelper**. This provides Active Directory integration services, including database instance registration.
* **Microsoft Search**. This provides full text search capabilities. This service must always run under the local system account.

Only the MSSQLSERVER database engine is required. The remaining services provide additional functionality and are required only in specific scenarios. Disable these services if they are not required.

**Note**   SQL Server should not be configured to run as the local System account or any account that is a member of the local Administrators group. For details about configuring the service account used to run MSSQLSERVER, see "[Step 4. Accounts](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_012)."

Disable the Microsoft DTC (if not required)

If you do not use distributed transactions through the Microsoft DTC, disable the service.

Step 3. Protocols

By preventing the use of unnecessary protocols, you reduce the surface area of attack. Configure SQL Server to support only clients that connect using the TCP/IP protocol. Disable all other protocols, unless required.

In this step, you:

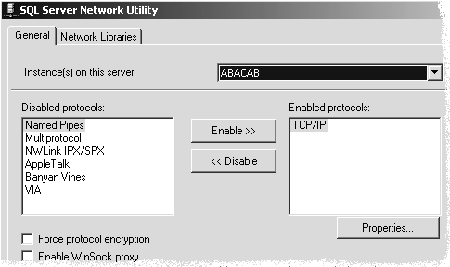
* **Restrict SQL Server to TCP/IP**.
* **Harden the TCP/IP Stack**.

Restrict SQL Server to TCP/IP

By enforcing the use of TCP/IP you can control who connects to the server on specific ports using IPSec policies or TCP/IP filtering. To support IPSec or TCP/IP filtering, your SQL Server should support client connections over TCP/IP only.

**To configure SQL Server network protocol support**

1. In the Microsoft SQL Server programs group, start the Server Network Utility.
2. Make sure that **TCP/IP** is the only SQL Server protocol that is enabled as shown in Figure 18.3. Disable all other protocols.



**Figure 18.3**

***Disabling all protocols except TCP/IP in the SQL Server Network Utility***

Harden the TCP/IP Stack

Windows 2000 and Windows Server 2003 allow you to control many parameters to configure TCP/IP implementation. Some of the defaults are geared toward server availability and specific features.

For information about how to harden the TCP/IP stack, see "[How To: Harden the TCP/IP Stack](https://msdn.microsoft.com/en-us/library/aa302363.aspx)" in the "How To" section of this guide.

Additional Considerations

To further improve your database server security, disable NetBIOS and SMB. Both protocols can be used to glean host configuration information, so you should remove them when possible. For more information about removing NetBIOS and SMB, see "Step 4. Protocols" in Chapter 16, "Securing Your Web Server."

Also consider using IPSec to restrict the ports on which your database server accepts incoming connections. For more information about how to do this, see "[How To: Use IPSec for Filtering Ports and Authentication](https://msdn.microsoft.com/en-us/library/aa302366.aspx)" in the "How To" section of this guide.

Step 4. Accounts

Follow the principle of least privilege for the accounts used to run and connect to SQL Server to restrict the capabilities of an attacker who manages to execute SQL commands on the database server. Also apply strong password policies to counter the threat of dictionary attacks.

In this step, you:

* **Secure the SQL Server service account**.
* **Delete or disable unused accounts**.
* **Disable the Windows guest account**.
* **Rename the administrator account**.
* **Enforce strong password policy**.
* **Restrict remote logins**.
* **Disable null sessions (anonymous logons)**.

Secure the SQL Server Service Account

Run the SQL Server service using a least privileged account to minimize the damage that can be done by an attacker who manages to execute operating system commands from SQL Server. The SQL Server service account should not be granted elevated privileges such as membership to the Administrators group.

**To create a new account to run the SQL Server service**

1. Start the Computer Management tool, and then expand **Local Users and Groups**.
2. Right-click the **Users** folder, and then click **New User**.
3. Create a new user making sure you use a strong password.

In the **New User** dialog box, clear the **User must change password at next logon** check box, and then select the **User cannot change password** and **Password never expires**check boxes.

1. Remove the new account from the **Users** group because this group is granted liberal access across the computer.

You can now configure SQL Server to run using this new account. For more information, see "[Step 10. SQL Server Security](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_018)."

Accessing the Network from SQL Server

If you need to access network resources from SQL Server, for example to perform network backups, for replication or log shipping, the SQL Server service account must be capable of being authenticated across the network. You have two choices. Either create a duplicate local account with the same name and password on the remote server, or use a least privileged domain account.

Delete or Disable Unused Accounts

Unused accounts and their privileges may be a haven for an attacker who has gained access to a server. Audit local accounts on the server and delete those that are unused. The recommendation is to first disable an account to see if this causes any problems before deleting the account, because deleted accounts cannot be recovered. Note that the **administrator** account and **guest** account cannot be deleted.

**Note**   During SQL Server 200 SP3 installation, Sqldbreg2.exe creates the **SQL Debugger** account. Visual Studio .NET uses this account when debugging stored procedures from managed .NET code. Because this account is only used to support debugging, you can delete it from production database servers.

Disable the Windows Guest Account

The Windows **guest** account is the account used when an anonymous connection is made to the computer. To restrict anonymous connections to your database server, keep this account disabled. By default, the **guest** account in Windows 2000 and Windows Server 2003 is disabled. To see if it is enabled, display the **Users** folder in the Computer Management tool. It is represented by a cross icon. If it isn't disabled, display its **Properties** dialog box and select the **Account is disabled** check box.

Rename the Administrator Account

The default local **administrator** account is a target for malicious use because of its high privileges on the computer. To improve security, rename the default **administrator** account and assign it a strong password.

Enforce Strong Password Policy

To counter password guessing and brute force dictionary attacks, apply strong password policies by configuring security policy. The keys to strong account and password policies are:

* **Set password length and complexity**. Enforcing strong passwords reduces the chance of successful password guessing or dictionary attacks.
* **Set password expiration**. Regularly expiring passwords reduces the chance that an old password will be used for unauthorized access. The expiration period is typically guided by a company's security policy.

Table 18.3 shows the default and recommended password policy settings.

**Table 18.3   Password Policy Default and Recommended Settings**

|  |  |  |
| --- | --- | --- |
| **Password Policy** | **Default Setting** | **Recommended Minimum Setting** |
| Enforce password history | 1 password remembered | 24 passwords remembered |
| Maximum password age | 42 days | 42 days |
| Minimum password age | 0 days | 2 days |
| Minimum password length | 0 characters | 8 characters |
| Passwords must meet complexity requirement | Disabled | Enabled |
| Strong password using reversible encryption for all users in the domain | Disabled | Disabled |

Additionally, log failed login attempts to detect and trace malicious behavior. For more information, see "[Step 9. Auditing and Logging](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_017)."

For more information about password policies, see password "Best Practices" on at [http://www.microsoft.com/resources/documentation/WindowsServ/2003/enterprise/proddocs/en-us/Default.asp?url=/resources/documentation/WindowsServ/2003/enterprise/proddocs/en-us/windows\_password\_protect.asp](https://technet.microsoft.com/en-us/library/cc706993(ws.10).aspx).

Restrict Remote Logons

Use the Local Security Policy tool to remove the "Access this computer from the network" user right from the Everyone group to restrict who can log on to the server remotely.

Disable Null Sessions (Anonymous Logons)

To prevent anonymous access, disable null sessions. These are unauthenticated or anonymous sessions established between two computers. Unless null sessions are disabled, an attacker can connect to your server anonymously, that is, without requiring authentication.

As soon as an attacker establishes a null session, a variety of attacks can be performed, including enumeration used to obtain system-related information from the target computer. The type of information that can be returned over a null session includes domain and trust details, shares, user information including groups and user rights, registry keys, and more. Disable them because they represent a significant security threat.

Restrict null sessions by setting RestrictAnonymous=1 in the registry at the following location.

HKLM\System\CurrentControlSet\Control\LSA\RestrictAnonymous=1

For more information, see Microsoft Knowledge Base article 246261, "How To: Use the RestrictAnonymous Registry Value in Windows 2000."

Additional Considerations

Consider the following steps to improve security for your database server:

* **Require approval for account delegation**. Do not mark domain accounts as trusted for delegation in Active Directory without special approval.
* **Do not use shared accounts**. Do not create shared account for use by multiple individuals. Give authorized individuals their own accounts. The activities of individuals can be audited separately and group membership and privileges appropriately assigned.
* **Restrict the local Administrators group membership**. Ideally, have no more than two administration accounts. This helps provide accountability. Also, do not share passwords, again to provide accountability.
* **Limit the administrator account to interactive logins**. If you perform only local administration, you can restrict your **administrator** account to interactive logons by removing the "Access this computer from the network" user right to deny network logon rights. This prevents users (well intentioned or otherwise) from remotely logging on to the server using the administrator account. If a policy of local administration is too inflexible, implement secure remote administration.

For more information about remote administration, see "[Remote Administration](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_024)" later in this chapter.

* **Enable NTLMv2 authentication**. If client computers connect to your database server by using Windows authentication, you should configure your database server to use the strongest version of Windows authentication, which is NTLMv2.

**Note**   To support NTLMV2, clients must be running Windows 2000, Windows Server 2003, or Windows NT® operating system version 4.0 with Service Pack 4.

**To enable NTLMv2 authentication from the Local Security Policy Tool**

1. Expand Local Policies, select Security Options, and then double-click LAN Manager Authentication Level.
2. Select Send NTLMv2 response only\refuse LM & NTLM.

This is the most secure setting.

**Note**   This is equivalent to setting the HKLM\System\CurrentControlSet\Control\Lsa\LMCompatibilityLevel DWORD value to 5.

Step 5. Files and Directories

In addition to securing operating system files using ACLs, harden NTFS permissions to restrict access to SQL Server program files, data files, and log files together with system level tools. Additionally, the SQL Server service account should have access only to what it needs.

In this step, you:

* **Verify permissions on SQL Server install directories**.
* **Verify** **Everyone group does not have permissions to SQL Server files**.
* **Secure setup log files**.
* **Secure or remove tools, utilities, and SDKs**.

Verify Permissions on SQL Server Install Directories

Verify the permissions listed in Table 18.4 to the account the SQL Server service is running under. The location specified in parentheses is the default install location. This may vary for your installation.

**Table 18.4   NTFS Permissions for SQL Server Service Account**

|  |  |
| --- | --- |
| **Location** | **Permissions for SQL Service Account** |
| Install location (\Program Files\Microsoft SQL Server\MSSQL\) | Read and Execute List Folder Contents Read |
| Database file directory (.mdf, .ndf, .ldf files) (\Program Files\Microsoft SQL Server\MSSQL\Data) | Full Control |
| Error log file directory (\Program Files\Microsoft SQL Server\MSSQL\LOG) | Full Control |
| Backup file directory (\Program Files\Microsoft SQL Server\MSSQL\Backup) | Full Control |
| Job temporary file output directory (\Program Files\Microsoft SQL Server\MSSQL\Jobs) | Full Control |

If you use Enterprise Manager to set the SQL Server service account, it gives the account Full Control permissions on the SQL Server installation directory and all subfolders (\Program Files\Microsoft SQL Server\MSSQL\\*).

By removing write permissions on this folder and all subfolders, and then selectively granting full control to the data, error log, backup and job file directories, the new account cannot overwrite SQL Server binaries.

Verify Everyone Group Does Not Have Permissions for SQL Server Files

The Everyone group should not have access to the SQL Server file location (by default, \Program Files\Microsoft SQL Server\MSSQL) This is achieved by verifying the Everyone group is not granted access via an ACL and giving explicit full control to only the SQL Service account, the Administrators group, and the local system account.

Secure Setup Log Files

After installing SQL Server 2000 Service Pack 1 or 2, the system administrator or service account password may be left in the SQL installation directory. Use the Killpwd.exe utility to remove instances of passwords from the log files.

For information about obtaining and using this utility, see Microsoft Knowledge Base article 263968, "FIX: Service Pack Installation May Save Standard Security Password in File."

Secure or Remove Tools, Utilities, and SDKs

SDKs and resource kits should not be installed on a production database server. Remove them if they are. In addition:

* Ensure that access to powerful system tools and utilities, such as those contained in the \Program Files directory, is restricted.
* Debugging tools should not be available on the database server. If production debugging is necessary, you should create a CD that contains the necessary debugging tools.

Additional Considerations

To further improve your database server security:

* **Remove unused applications that may be installed on the server**. If you have applications on the server that you do not use, then remove them.
* **Encrypt your data files using Encrypting File System (EFS)**. You can use EFS to protect your data files. If your data files are stolen, encrypted files are more difficult to read. The use of EFS for SQL Server data files is supported.

When using EFS, you should be aware of the following:

* + Encrypt the database files (.MDF) and not the log files (.LDF). If you encrypt the log files, then SQL Server cannot open your database.
  + Encrypt at the file level, not the directory level. While it is often a best practice to encrypt at the directory level when using EFS so that when new files are added they are automatically encrypted, you should encrypt your SQL Server data files at the file level only. This avoids encrypting your log files.
  + Evaluate the performance cost. The use of EFS incurs a performance penalty. Test EFS before using it in your scenario to determine the actual performance impact. Usually the performance penalty is negligible because the data file is decrypted by SQL Server when the process starts.

To implement EFS, right-click the directory, click **Advanced**, and then click **Encrypt contents to be secure**. For more information about EFS, see the following resources:

* + Microsoft Knowledge Base article 23050, "How To: Encrypt Data Using EFS in Windows 2000."
  + Microsoft TechNet article, "[Step-by-Step Guide to Using the Encrypting File System](https://technet.microsoft.com/en-us/windowsserver/2003/default.aspx)."

Step 6. Shares

Remove any unused shares and harden the NTFS permissions on any required shares. By default, all users have full control on newly created file shares. Harden these default permissions to make sure that only authorized users can access files exposed by the share. Also, use NTFS ACLs on files and folders exposed by the share in addition to explicit share permissions.

In this step, you:

* **Remove unnecessary shares**.
* **Restrict access to required shares**.

Remove Unnecessary Shares

Remove all unnecessary shares. To review shares, start the Computer Management MMC snap-in and select **Shares** under **SharedFolders**.

Restrict Access to Required Shares

Remove the Everyone group and grant specific permissions instead. Everyone is used when you do not have restrictions on who has access to the share.

Additional Considerations

If you are not allowing remote administration of the computer, remove unused administrative shares, for example, C$ and Admin$.

**Note**   Some applications may require administrative shares such as Microsoft Management Server (SMS) or Microsoft Operations Manager (MOM). For more information, see Microsoft Knowledge Base article 318751, "How To: Remove Administrative Shares in Windows 2000 or Windows NT 4.0."

**Note**   Sharing is disabled by default in Windows Server 2003. If you enable sharing, you need to consider the recommendations above**.**

Step 7. Ports

By default, SQL Server listens on TCP port 1433 and uses UDP port 1434 for client-server negotiation. Use a combination of firewalls and IPSec policies to restrict access to these ports to minimize the avenues of attack open to an attacker.

In this step, you:

* **Restrict access to the SQL server port**.
* **Configure named instances to listen on the same port**.
* **Configure the firewall to support DTC traffic (if necessary)**.

Restrict Access to the SQL Server Port

Use a perimeter firewall to prevent direct access from the Internet to the SQL Server ports  by default, TCP port 1433 and UDP port 1434. This does not protect your server against internal attacks. Configure IPSec policies to limit access, through TCP port 1433 and UDP port 1434, from Web or application servers that connect to the database by design.

For more information, see "[How To: Use IPSec](https://msdn.microsoft.com/en-us/library/aa302366.aspx)" in the "How To" section of this guide.

Configure Named Instances to Listen on the Same Port

By default, named instances of SQL Server dynamically allocate a port number and use UDP negotiation with the client to allow the client to locate the named instance. To avoid opening a range of port numbers on the internal firewall or having to create multiple IPSec policies, use the Server Network Utility to configure the instance to listen on a specific port number.

If you reconfigure the port number on the server, you must also reconfigure any clients to make sure they connect to the correct port number. You might be able to use the Client Network Utility, but this utility should not be installed on a Web server. Instead, applications can specify the port number in their connection strings by appending the port number to either the Server or Data Source attributes as shown in the following code.

"Server=YourServer|YourServerIPAddress,PortNumber"

Configure the Firewall to Support DTC Traffic (if necessary)

If your applications use Enterprise Services (COM+) transactions and require the services of the DTC, you may have to specifically configure the firewall that separates your Web application and database server to allow DTC traffic between separate DTC instances and between the DTC and SQL Server.

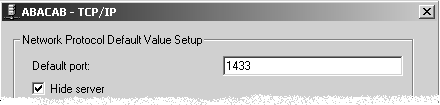
For more information about opening ports for the DTC, see Microsoft Knowledge Base article 250367, "INFO: Configuring Microsoft Distributed Transaction Coordinator (DTC) to Work Through a Firewall."

Additional Considerations

Consider using the **Hide** **Server** option from the Server Network Utility as shown in Figure 18.4. If you select this option in the TCP/IP properties dialog box in the SQL Network Utility, SQL Server is reconfigured to listen on port 2433. It also disables responses to broadcast requests from clients that try to enumerate SQL Server instances.

This measure cannot be relied upon to completely hide the SQL Server port. This is not possible because there are a variety of ways to enumerate ports to discover its location.

**Note**   This option can be used only if you have a single instance of SQL Server. For more information, see Microsoft Knowledge Base article 308091, "BUG: Hide Server Option Cannot Be Used on Multiple Instances of SQL Server 2000."



**Figure 18.4**

***Setting the Hide Server option from the Server Network Utility***

Step 8. Registry

When you install SQL Server, it creates a number of registry entries and subentries that maintain vital system configuration settings. It is important to secure these settings to prevent an attacker from changing them to compromise the security of your SQL Server installation.

When you install SQL Server, it creates the following registry entries and subentries:

* For a default instance:
* HKEY\_LOCAL\_MACHINE\SOFTWARE\MICROSOFT\MSSQLSERVER
* For a named instance:
* HKEY\_LOCAL\_MACHINE\SOFTWARE\MICROSOFT\MICROSOFT SQL SERVER\INSTANCENAME
* For the SQL service:
* HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\MSSQLSERVER

In this step, you:

* **Verify permissions for the SQL Server registry keys**.
* **Secure the SAM (stand-alone servers only)**.

Verify Permissions for the SQL Server Registry Keys

Use Regedt32.exe to verify the Everyone group does not have permissions on the SQL Server registry keys, listed above. The following controls are in place by default:

Administrators: Full Control

SQL Server service account: Full Control

**Note**   The Microsoft Baseline Security Analyzer will verify the registry permissions. Use the tool as an alternative to manually verifying the permissions with Regedt32.exe.

Secure the SAM (Stand-alone Servers Only)

Stand-alone servers store account names and one-way password hashes (LMHash) in the local SAM database, which is part of the registry. Generally, only members of the Administrators group have access to the account information.

Although the passwords are not actually stored in the SAM and password hashes are not reversible, if an attacker obtains a copy of the SAM database, he or she can use brute force password cracking techniques to obtain valid credentials.

Restrict LMHash storage in the SAM by creating the key (not value) NoLMHash in the registry as shown below.

HKLM\System\CurrentControlSet\Control\LSA\NoLMHash

For more information, see Microsoft Knowledge Base article 299656, "New Registry Key to Remove LM Hashes from Active Directory and Security Account Manager."

Step 9. Auditing and Logging

Auditing does not prevent system attacks, although it is a vital aid in identifying intruders, attacks in progress, and to diagnose attack footprints. It is important to enable all auditing mechanisms at your disposal, including Windows operating system level auditing and SQL Server login auditing. SQL Server also supports C2 level extended auditing. This may be required in specific application scenarios, where auditing requirements are stringent.

In this step, you:

* **Log all failed Windows login attempts**.
* **Log all failed actions across the file system**.
* **Enable SQL Server login auditing**.

Log All Failed Windows Logon Attempts

You must log failed Windows logon attempts to be able to detect and trace malicious behavior.

**To audit failed logon attempts**

1. Start the Local Security Policy tool.
2. Expand **Local Policies** and then select **Audit Policy**.
3. Double-click **Audit account logon events**.
4. Click **Failure**, and then click **OK**.

Windows logon failures are recorded as events in the Windows security event log. The following event IDs are suspicious:

* **531**. This means an attempt was made to log on using a disabled account.
* **529**. This means an attempt was made to log on using an unknown user account or using a valid user account but with an invalid password. An unexpected increase in the number of these audit events might indicate an attempt to guess passwords.

Log All Failed Actions Across the File System

Use NTFS auditing on the file system to detect potentially malicious attempts. This is a two-step process:

**To enable logging**

1. Start the Local Security Policy tool.
2. Expand **Local Policies**, and then select **Audit Policy**.
3. Double click **Audit object access**.
4. Click **Failure**, and then click **OK**.

**To audit failed actions across the file system**

1. Start Windows Explorer and navigate to the root of the file system.
2. Right-click the root of the file system, and then click **Properties**.
3. Click the **Security** tab.
4. Click **Advanced**, and then click the **Auditing** tab.
5. Click **Add**, and then enter **Everyone** into the **object name to select** field.
6. Click **OK**, and then select the **Full Control** check box in the **Failed** column to audit all failed events.

By default, this applies to the current folder and all subfolders and files.

1. Click **OK** three times to close all open dialog boxes.

Failed audit events are logged to the Windows security event log.

Enable SQL Server Login Auditing

By default, SQL Server login auditing is not enabled. Minimally, you should audit failed logins. Auditing failed login attempts is a useful way of detecting an attacker who is trying to crack account passwords. For more information, about how to enable SQL Server auditing, see "[Step 10. SQL Server Security](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_018)."

Additional Considerations

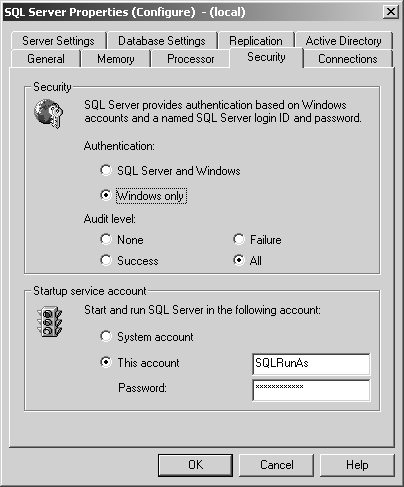
The following are additional measures to consider when auditing and logging:

* **Consider shutting down the system if unable to log security audits**. This policy option is set in the **Security Options** of the **Local Security Settings** management console. Consider this setting for highly secure servers.
* **Consider C2 level auditing**. SQL Server offers an auditing capability that complies with the U.S. Government C2 certification. C2 level auditing provides substantially more audit information at the expense of increased disk storage requirements.

For more information about the configuration of a C2-compliant system, see the TechNet article "SQL Server 2000 C2 Administrator's and User's Security Guide" at [http://technet.microsoft.com/en-us/library/cc966496.aspx](https://technet.microsoft.com/en-us/library/cc966496.aspx).

Step 10. SQL Server Security

The settings discussed in this section are configured using the **Security** tab of the **SQL** **Server Properties** dialog box in Enterprise Manager. The settings apply to all the databases in a single instance of SQL Server. The **SQL Server Properties** dialog box is shown in Figure 18.5.



**Figure 18.5**

***SQL Server security properties***

In this step, you:

* **Set SQL Server authentication to Windows only**.
* **Set SQL Server audit level to Failure or All**.
* **Run SQL Server using a least privileged account**.

Set SQL Server Authentication to Windows Only

You should configure SQL Server to support Windows-only authentication because it provides a number of benefits. Credentials are not passed over the network, you avoid embedding usernames and passwords in database connection strings, security is easier to manage because you work with the single Windows security model instead of a separate SQL Server security model, and login security improves through password expiration periods, minimum lengths, and account lockout policies.

**To configure Windows only authentication**

1. Start SQL Server Enterprise Manager, expand the **SQL Server Group**, and then expand your SQL Server.
2. Right-click your SQL Server, and then click **Properties**.
3. Click the **Security** tab.
4. Select **Windows only**, and then click **OK**.
5. Restart SQL Server for the changes to take effect.

Set SQL Server Audit Level to Failure or All

By default, SQL Server login auditing is not enabled. Minimally, you should audit failed logins.

**Note**   Log entries are written to SQL log files. By default, these are located in C:\Program Files\Microsoft SQL Server\MSSQL\LOG. You can use any text reader, such as Notepad, to view them.

**To enable SQL Server auditing**

1. Start **SQL Server Enterprise Manager**, expand the **SQL Server Group**, and then expand your SQL Server.
2. Right-click your SQL Server, and then click **Properties**.
3. Click the **Security** tab.
4. Set the Audit level to either **All** or **Failure**.
5. Restart SQL Server for the changes to audit policy to take effect.

For more information about SQL Server audit logs, see the TechNet article and its section "Understanding the Audit Log" in the "SQL Server 2000 Auditing" article at [http://technet.microsoft.com/en-us/library/dd277388.aspx](https://technet.microsoft.com/en-us/library/dd277388.aspx).

Run SQL Server Using a Least Privileged Account

Run the SQL Server service using a least privileged account to minimize the damage that can be done by an attacker who manages to execute operating system commands from SQL Server. The SQL Server service account should not be granted elevated privileges such as membership to the Administrators group.

**To configure the SQL Server run as account**

This procedure uses Enterprise Manager instead of the Services MMC snap-in because Enterprise Manager automatically grants the user rights that a SQL Server service account requires.

1. Start **SQL Server Enterprise Manager**, expand the **SQL Server Group**, and then expand your SQL Server.
2. Right-click your SQL Server, and then click **Properties**.
3. Click the **Security** tab.
4. Click **This account** in the Startup service account group. Enter the user name and password of your least privileged account.
5. Restart SQL Server for the changes to take effect.

**Note**   If you use the SQLSERVERAGENT service, the run-as account must also be changed. Use the Services MMC snap-in to change this setting.

For more information about creating a least privileged account to run SQL Server, see "[Step 4. Accounts](https://msdn.microsoft.com/en-us/library/ff648664.aspx#c18618429_012)."

Step 11. SQL Server Logins, Users, and Roles

To be able to access objects in a database you need to pass two layers of security checks. First, you need to present a valid set of login credentials to SQL Server. If you use Windows authentication, you need to connect using a Windows account that has been granted a SQL Server login. If you use SQL Server authentication, you need to supply a valid user name and password combination.

The login grants you access to SQL Server. To access a database, the login must be associated with a database user inside the database you want to connect to. If the login is associated with a database user, the capabilities of the login inside the database are determined by the permissions associated with that user. If a login is not associated with a specific database user, the capabilities of the login are determined by the permissions granted to the public role in the database. All valid logins are associated with the public role, which is present in every database and cannot be deleted. By default, the public role within any database that you create is not granted any permissions.

Use the following recommendations to improve authorization settings in the database:

* **Use a strong sa (system administrator) password**.
* **Remove the SQL guest user account**.
* **Remove the BUILTIN\Administrators server login**.
* **Do not grant permissions for the public role**.

Use a Strong sa (System Administrator) Password

The default system administrator (**sa**) account has been a subject of countless attacks. It is the default member of the SQL Server administration fixed server role **sysadmin**. Make sure you use a strong password with this account.

**Important**   The **sa** account is still active even when you change from SQL authentication to Windows authentication.

Apply strong passwords to all accounts, particularly privileged accounts such as members of the **sysadmin** and **db\_owner** roles. If you are using replication, also apply a strong password to the **distributor\_admin** account that is used to establish connections to remote distributor servers.

Remove the SQL Guest User Account

When you install SQL Server, a **guest** user account is created if the Windows 2000 **guest** account is enabled. A login assumes the identity of **guest** if the login has access to SQL Server but does not have access to a database through a database user account.

It is a good idea to disable the Windows **guest** account. Additionally, remove the **guest** account from all user-defined databases. Note that you cannot remove **guest** from the master, tempdb, and replication and distribution databases.

Remove the BUILTIN\Administrators Server Login

By default, the BUILTIN\Administrators local Windows group is added to the **sysadmin** fixed server role to administer SQL Server. This means that domain administrators who are members of BUILTIN\Administrators have unrestricted access to the SQL Server database. Most companies differentiate the role of domain administrator and database administrator. If you do this*,* remove the BUILTIN\Administrators SQL Server login. It is a good idea to create a specific Windows group containing specific database administrations in its place and added to SQL server as a server login as shown in the following procedure.

**To add a new login for database administrators**

1. Start Enterprise Manager.
2. Expand**Microsoft SQL Server**, expand **SQL Server Group**, and then expand your SQL Server.
3. Expand the Security folder, select and right-click **Logins**, and then click **New** **Login**.
4. In the **Name** field, enter a custom Windows group that contains only database administrators.
5. Click the **Server Roles** tab, and then select **System Administrators**.

This adds the new login to the **sysadmin** server role.

**To delete the BUILTIN\Administrators login**

1. Start Enterprise Manager.
2. Expand **Microsoft SQL Server**, expand **SQL Server Group**, and then expand your SQL Server.
3. Expand the **Security** folder, and select **Logins**. If BUILTIN\Administrators appears in the list of logins, right-click it, and then click **Delete** to remove the login.

For more information about reconfiguring the SQL service accounts after the installation, see the MSDN article, "Changing Passwords and User Accounts" at [http://msdn.microsoft.com/en-us/library/aa197070(SQL.80).aspx](https://msdn.microsoft.com/en-us/library/aa197070(sql.80).aspx).

Do Not Grant Permissions for the Public Role

All databases contain a public database role. Every other user, group, and role is a member of the public role. You cannot remove members of the public role. Instead, do not grant the permissions for the public role that grant access to your application's database tables, stored procedures, and other objects. Otherwise, you cannot get the authorization that you want using user-defined database roles because the public role grants default permissions for users in a database.

Additional Considerations

Also consider the following recommendations when configuring SQL Server logins, users, and roles:

* **Limit the members of sysadmin**. To make sure there is individual accountability, restrict the number of accounts that are members of the **sysadmin** role. Ideally, no more than two users are members of this role.
* **Grant restricted database permissions**. Assign accounts only the absolute minimum permissions required to do a job. Avoid using the built-in roles, such as **db\_datareader** and **db\_datawriter**. These roles do not provide any authorization granularity and these roles have access to all of your custom database objects.
* **Do not change the default permissions that are applied to SQL Server objects**. In versions of SQL Server earlier than Service Pack 3, the public role does have access to various default SQL Server database objects. With Service Pack 3, the security design has been reviewed and security has been improved by removing the public role where it is unnecessary and by applying more granular role checks.

Step 12. SQL Server Database Objects

SQL Server provides two sample databases for development and education together with a series of built-in stored procedures and extended stored procedures. The sample databases should not be installed on production servers and powerful stored procedures and extended stored procedures should be secured.

In this step, you:

* **Remove the sample databases**.
* **Secure stored procedures**.
* **Secure extended stored procedures**.
* **Restrict cmdExec access to the sysadmin role**.

Remove the Sample Databases

Use SQL Server Enterprise Manager to remove any sample databases. By default SQL Server includes the Pubs and Northwind sample databases.

Secure Stored Procedures

Restrict access to your application's stored procedures. Do not grant the public role or the **guest** user access to any stored procedures that you create. Your main line of defense for securing stored procedures is to ensure that you use strong authentication, and then to provide granular authorization, allowing only the necessary users permission to run the stored procedures.

The recommended approach is to create a SQL Server login for your application, map the login to a database user, add the user to a user-defined database role, and then grant permissions to the role.

Secure Extended Stored Procedures

Deleting stored procedures is not tested and not supported.

Restrict cmdExec Access to the sysadmin Role

The **cmdExec** function is used by the SQL Server Agent to execute Windows command-line applications and scripts that are scheduled by the SQL Server Agent. Prior to SQL Server Service Pack 3, by default the SQL Server Agent allows users who are not in the **sysadmin** role to schedule jobs that may require privileged access to the system. You should change this setting to allow members only of the **sysadmin** role to schedule jobs.

**To restrict cmdExec access to the sysadmin role**

1. Start **SQL Server Enterprise Manager**, expand the **SQL Server Group**, and then expand your **SQL Server**.
2. Expand the **Management**node**,** right-click **SQL Server Agent,** and then click **Properties**.

The **SQL Server Agent Properties**dialog box is displayed.

1. Click the **Job System** tab.
2. At the bottom of the dialog, select the **Only users with SysAdmin privileges can execute CmdExec and ActiveScripting job steps** check box.
3. Click **OK**.

**Note**   This change may require you to supply a user name and password. If the SQL Server service account is least privileged user (as advocated earlier in this chapter), you will be prompted for the user name and password of an administrator account that has privileges to modify the service.

Snapshot of a Secure Database Server

When you have a snapshot view that shows the attributes of a secured SQL Server database server, you can quickly and easily compare settings with your own server. The settings shown in Table 18.5 are based on an analysis of SQL Server database servers that have proven to be very resilient to attack and demonstrate sound security practices.

**Table 18.5   Snapshot of a Secure Database Server**

|  |  |
| --- | --- |
| **Component** | **Characteristics** |
| Patches and Updates | The latest service packs and patches are applied for Windows 2000 / Windows Server 2003 and SQL Server. |
| Services | Nonessential services are disabled.  The MSDTC is disabled if not used.  The MSSearch service is disabled if not required.  The SQLServerAgent service is disabled if not required.  The MSSQLServerADHelper service is disabled if not required. |
| Protocols | Unnecessary protocols are removed or disabled.  The following protocols are not enabled on the server: NetBIOS and SMB.  The TCP/IP stack is hardened. |
| Accounts | SQL Server service account is secured (least privileged).  Unnecessary Windows accounts are deleted or disabled.  The Windows **guest** account is disabled.  A new administrator account is created.  Strong password policy is enforced.  Remote logons are restricted.  Null sessions (anonymous logons) are disabled.  Approval is required for account delegation.  Shared accounts are not used.  Membership of the local Administrators group is limited (ideally, no more than two members).  The administrator account is limited to interactive logins (or a secure remote administration solution is provided).  NTLMv2 authentication is enabled and enforced (LMCompatibilityLevel is set to 5). |
| Files and Directories | Volumes are formatted with NTFS.  **Everyone** group has no rights to system or tools directories.  Samples directories, Help directories, and unused admin directories are removed from the server.  Permissions are hardened on SQL Server installation folder.  Passwords removed from Service Pack 1 and Service Pack 2 setup log files.  Tools, utilities and SDKs are removed.  Unused applications are removed.  Sensitive data files are encrypted using EFS. (This is optional for database files (.mdf), but not for log files (.ldf)). |
| Shares | Unnecessary shares are removed from the server.  Access is restricted to required shares.  Shares are not accessible by **Everyone**, unless necessary.  Administration shares (C$, Admin$) are removed if they are not required. |
| Ports | All ports except SQL Server listening port [Default 1433] are blocked  Named instances are configured to listen on the same port.  A non-standard SQL Server port (not TCP 1443) is used as an additional layer of defense.  The hide server option is used as an additional layer of defense (optional).  The firewall is configured to support DTC traffic (if necessary).  A firewall is used to separate users from the SQL TCP/IP port. |
| Registry | **Everyone** group is removed from SQL Server registry keys.  SAM is secured (stand-alone servers only). |
| Auditing and Logging | Failed Windows logon attempts are logged.  Failed actions across the file system are logged.  SQL Server login auditing is enabled. |
| **SQL Server Settings** | |
| SQL Server Security | Authentication setting for SQL Server is Windows Only if possible.  SQL Server audit level set to Failure or All.  The SQL Server Startup Service account is a least privileged account. |
| SQL Server Logins, Users and Roles | The **sa** account has a strong password.  SQL Server **guest** accounts are removed from non-system databases.  The BUILTIN\Administrators group is removed from the SQL Server logins.  The sysadmin role does not contain the BUILTIN\Administrators group.  Permissions are not granted for the public role.  The sysadmin role contains no more than two users.  Restricted (granular) database permissions are granted (Built-in, non-granular roles such as db\_datareader and db\_datawriter are avoided)  Default permissions for SQL Server objects are not changed. |
| SQL Server Database Objects | All sample databases are removed from the server.  Stored procedures are secured.  Extended stored procedures are secured.  cmdExec is restricted to the sysadmin role only. |

Additional Considerations

In addition to the steps described in this chapter, consider the following guidelines:

* **Install a certificate on the database server**. If you use Windows authentication (NTLM or Kerberos), logon credentials are not passed over the network to SQL Server. If you use SQL authentication, it is a good idea to secure the credentials because they are passed to SQL Server in unencrypted format. Do this by installing a certificate on the database server. This automatically results in the encryption of SQL credentials over the wire. It is also a good idea to make sure that your application securely stores database connection strings. For more information, see Chapter 14, "[Building Secure Data Access](https://msdn.microsoft.com/en-us/library/aa302430.aspx)."
* **Restrict access to sensitive commands and stored procedures**. SQL Server provides powerful hooks into the operating system. For example, you can use the **xp\_cmdshell** extended stored procedure to run any operating system command. If an attacker manages to run arbitrary commands in the database, for example through a SQL injection vulnerability, the ability to execute operating system commands is limited only by the security credentials of the account used to run SQL Server. This is the primary reason for running SQL Server with a least privileged local account.
* **Use a dedicated computer as a database server**. Also cluster it for failover.
* **Physically protect the database server**. Locate the server in a secure computer room.
* **Restrict local logons**. Do not allow anyone to locally log on to the server, apart from the administrator.

Staying Secure

You need to regularly monitor the security state of your database server and update it regularly to help prevent newly discovered vulnerabilities from being exploited. To help keep your database server secure:

* **Perform regular backups**.
* **Audit group membership**.
* **Monitor audit logs**.
* **Stay current with service packs and patches**.
* **Perform security assessments**.
* **Use security notification services**.

Perform Regular Backups

You must be able to restore data in the event of a compromise. If you have a recovery system in place, test it before you actually need it. The first time you need to recover data should not be the first time you test your backup and restore process. For more information on backing up and restoring SQL Server, see the following resources:

* SQL Server 2000 documentation, "Backing Up and Restoring Databases"
* "Backup and Restore Strategies with SQL Server 2000," by Rudy Lee Martinez, [http://www1.us.dell.com/content/topics/global.aspx/power/en/ps4q00\_martin?c=us&cs=555&l=en&s=biz](http://www.dell.com/content/topics/global.aspx/power/en/ps4q00_martin?c=us&cs=555&l=en&s=biz)

Audit Group Membership

Keep track of user group membership, particularly for privileged groups such as Administrators. The following command lists the members of the Administrators group:

**net localgroup administrators**

Monitor Audit Logs

Monitor audit logs regularly and analyze the log files by manually viewing them or use the technique described in Microsoft Knowledge Base article 296085, "How To: Use SQL Server to Analyze Web Logs."

Stay Current with Service Packs and Patches

Set up a schedule to analyze your server's software and subscribe to security alerts. Use MBSA to regularly scan your server for missing patches. The following links provide the latest updates:

* **Windows 2000 service packs**. The latest service packs are listed at [http://technet.microsoft.com/en-us/windowsserver/2000/default.aspx](https://technet.microsoft.com/en-us/windowsserver/2000/default.aspx).
* **Windows 2003 service packs.** The latest service packs are listed at [http://technet.microsoft.com/en-us/windowsserver/bb463273.aspx](https://technet.microsoft.com/en-us/windowsserver/bb463273.aspx).
* **Critical updates**. These updates help to resolve known issues and help protect your computer from known security vulnerabilities. For the latest critical updates, see [http://technet.microsoft.com/en-us/windowsserver/2000/default.aspx](https://technet.microsoft.com/en-us/windowsserver/2000/default.aspx).
* **Advanced** **security updates**. Also monitor the advanced security updates or Windows 2000 at [http://technet.microsoft.com/en-us/windowsserver/2000/default.aspx](https://technet.microsoft.com/en-us/windowsserver/2000/default.aspx).

These also help protect your computer from known security vulnerabilities.

Perform Security Assessments

Use MBSA to regularly check for security vulnerabilities and to identify missing patches and updates. Schedule MBSA to run daily and analyze the results to take action as needed. For more information about automating MBSA, see "[How To: Use the Microsoft Baseline Security Analyzer](https://msdn.microsoft.com/en-us/library/aa302360.aspx)" in the "How To" section of this guide.

Use Security Notification Services

Use the Microsoft services listed in Table 18.6 to obtain security bulletins with notifications of possible system vulnerabilities.

**Table 18.6   Security Notification Services**

|  |  |
| --- | --- |
| **Service** | **Location** |
| TechNet security Web site | [http://www.microsoft.com/technet/security/current.aspx](https://www.microsoft.com/technet/security/current.aspx)  Use this Web page to view the security bulletins that are available for your system. |
| Microsoft Security Notification Service | [http://register.microsoft.com/subscription/subscribeme.asp?ID=135](https://technet.microsoft.com/en-us/security/dd252948.aspx)  Use this service to register for regular email bulletins that notify you of the availability of new fixes and updates |

Additionally, subscribe to the industry security alert services shown in Table 18.7. This allows you to assess the threat of a vulnerability where a patch is not yet available.

**Table 18.7   Industry Security Notification Services**

|  |  |
| --- | --- |
| **Service** | **Location** |
| CERT Advisory Mailing List | <http://www.cert.org/contact_cert/certmaillist.html>  Informative advisories are sent when vulnerabilities are reported. |
| Windows and .NET Magazine Security UPDATE | [Content link no longer available, original URL:"http://email.winnetmag.com/winnetmag/winnetmag\_prefctr.asp"]  Announces the latest security breaches and identifies fixes. |
| NTBugtraq | [http://www.ntbugtraq.com/default.asp?pid=31&sid=1#020](http://www.ntbugtraq.com/default.aspx?pid=31&sid=1)  This is an open discussion of Windows security vulnerabilities and attacks. Vulnerabilities which currently have no patch are discussed. |

Remote Administration

Administrators often need to be able to administer multiple servers. Make sure the requirements of your remote administration solution do not compromise security. If you need remote administration capabilities, the following recommendations help improve security:

* **Restrict the number of administration accounts**. This includes restricting the number of administration accounts as well as restricting which accounts are allowed to logon remotely.
* **Restrict the tools**. The main options include SQL Enterprise Manager and Terminal Services. Both SQL Enterprise Manager and Terminal Services use Windows security. As such, the main considerations here are restricting the Windows accounts and the ports you use.
* **Restrict the computers that are allowed to administer the server**. IPSec can be used to restrict which computers can connect to your SQL Server.

Securing Terminal Services

It is possible to use Microsoft Terminal Services securely to remotely administer your database server.

Terminal Services is based on Microsoft's proprietary protocol known as Remote Desktop Protocol (RDP). RDP uses the TCP 3389 port and supports two concurrent users. The following sections describe how to install and configure Terminal Services for secure administration:

* **Install Terminal Services**.
* **Configure Terminal Services**.

Install Terminal Services

**To install terminal services, do the following**

1. Install Terminal Services by using **Add/Remove Programs** from the Control Panel. Use the **Add/Remove Windows Components** option. You do not need to install the Terminal Services Licensing service for remote administration.
2. Configure Terminal Services for remote administration mode.
3. Remove the **TsInternetUser** user account from the system, which is created during Terminal Services installation. This account is used to support anonymous Internet access to Terminal Services, which should not be enabled on the server.

Configure Terminal Services

**Use the Terminal Services configuration MMC snap-in available from the Administrative Tools program group to configure the following**

1. There are three levels (Low, Medium, and High) of encryption available for connections to Terminal Services. Set the encryption to 128-bit key. Note that the Windows high encryption pack should be installed on both the server and the client.
2. Configure the Terminal Services session to disconnect after idle connection time limit. Set it to end a disconnected session. A session is considered to be disconnected if the user closes the Terminal Services client application without logging off in a period of 10 minutes.
3. Finally, restrict permissions to access Terminal Services. Use the RDP permissions tab in the RDP dialog box. By default, all members of the Administrators group are allowed to access Terminal Services. If you do not want all members of the Administrators group to access Terminal Services, remove the group and add individual accounts that need access. Note that the **SYSTEM** account must be in the list.

Use a secure VPN connection between the client and the server or an IPSec tunnel for enhanced security. This approach provides mutual authentication and the RDS payload is encrypted.

Copying Files over RDP

Terminal Services does not provide built-in support for file transfer. However, you can install the File Copy utility from the Windows 2000 Server Resource Kit to add file transfer functionality to the clipboard redirection feature in Terminal Services. For more information about the utility and installation instructions, see Microsoft Knowledge Base article 244732, "How To: Install the File Copy Tool Included with the Windows 2000 Resource Kit."

Summary

Database servers are a prime target for attackers. The database server must be secured against internal, external, network level, and application level attacks. A secure database server includes a hardened SQL Server 2000 installation on top of a hardened Windows 2000 / Windows Server 2003 installation, coupled with secure network defenses provided by routers and firewalls.

For a quick reference checklist, see "[Checklist: Securing Your Database Server](https://msdn.microsoft.com/en-us/library/aa302343.aspx)" in the "Checklists" section of this guide.

Additional Resources

For more information about SQL Server security, see the following resources:

* Microsoft SQL Server Security home page: [Content link no longer available, original URL:"http://www.microsoft.com/sql/techinfo/administration/2000/security/default.asp"] http://www.microsoft.com/sql/techinfo/administration/2000/security.asp.
* SQL Server 2000 Resource Kit CD, Chapter 10  Implementing Security: [http://technet.microsoft.com/en-us/library/cc917634.aspx](https://technet.microsoft.com/en-us/library/cc917634.aspx).
* "SQL Server 2000 Security Model," by Richard Waymire and Ben Thomas: [http://technet.microsoft.com/en-us/library/cc966453.aspx](https://technet.microsoft.com/en-us/library/cc966453.aspx).
* For information about changing the SQL Server service account, see Microsoft Knowledge Base article 283811, "How To: Change the SQL Server Service Account Without Using SQL Enterprise Manager in SQL Server 2000."
* For information about SQL Server auditing, see the TechNet article, "SQL Server 2000 Auditing," by John Howie, at [http://technet.microsoft.com/en-us/library/dd277388.aspx](https://technet.microsoft.com/en-us/library/dd277388.aspx).