# SQL

## **Applies to:**  SQL Server (all supported versions)

Securing SQL Server can be viewed as a series of steps, involving four areas: the platform, authentication, objects (including data), and applications that access the system. The following topics will guide you through creating and implementing an effective security plan.

You can find more information about SQL Server security at the [SQL Server](https://go.microsoft.com/fwlink/?LinkID=31629) Web site. This includes a best practice guide and a security checklist. This site also contains the latest service pack information and downloads.

## Platform and Network Security

The platform for SQL Server includes the physical hardware and networking systems connecting clients to the database servers, and the binary files that are used to process database requests.

## Physical Security

Best practices for physical security strictly limit access to the physical server and hardware components. For example, use locked rooms with restricted access for the database server hardware and networking devices. In addition, limit access to backup media by storing it at a secure offsite location.

Implementing physical network security starts with keeping unauthorized users off the network. The following table contains more information about networking security information.

| **For information about** | **See** |
| --- | --- |
| SQL Server Compact and network access to other SQL Server editions | "Configuring and Securing the Server Environment" in SQL Server Compact Books Online |

## Operating System Security

Operating system service packs and upgrades include important security enhancements. Apply all updates and upgrades to the operating system after you test them with the database applications.

Firewalls also provide effective ways to implement security. Logically, a firewall is a separator or restrictor of network traffic, which can be configured to enforce your organization's data security policy. If you use a firewall, you will increase security at the operating system level by providing a chokepoint where your security measures can be focused. The following table contains more information about how to use a firewall with SQL Server.

| **For information about** | **See** |
| --- | --- |
| Configuring a firewall to work with SQL Server | [Configure a Windows Firewall for Database Engine Access](https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-a-windows-firewall-for-database-engine-access?view=sql-server-ver16) |
| Configuring a firewall to work with Integration Services | [Integration Services Service (SSIS Service)](https://learn.microsoft.com/en-us/sql/integration-services/service/integration-services-service-ssis-service?view=sql-server-ver16) |
| Configuring a firewall to work with Analysis Services | [Configure the Windows Firewall to Allow Analysis Services Access](https://learn.microsoft.com/en-us/analysis-services/instances/configure-the-windows-firewall-to-allow-analysis-services-access) |
| Opening specific ports on a firewall to enable access to SQL Server | [Configure the Windows Firewall to Allow SQL Server Access](https://learn.microsoft.com/en-us/sql/sql-server/install/configure-the-windows-firewall-to-allow-sql-server-access?view=sql-server-ver16) |
| Configuring support for Extended Protection for Authentication by using channel binding and service binding | [Connect to the Database Engine Using Extended Protection](https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/connect-to-the-database-engine-using-extended-protection?view=sql-server-ver16) |

Surface area reduction is a security measure that involves stopping or disabling unused components. Surface area reduction helps improve security by providing fewer avenues for potential attacks on a system. The key to limiting the surface area of SQL Server includes running required services that have "least privilege" by granting services and users only the appropriate rights. The following table contains more information about services and system access.

| **For information about** | **See** |
| --- | --- |
| Services required for SQL Server | [Configure Windows Service Accounts and Permissions](https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions?view=sql-server-ver16) |

If your SQL Server system uses Internet Information Services (IIS), additional steps are required to help secure the surface of the platform. The following table contains information about SQL Server and Internet Information Services.

| **For information about** | **See** |
| --- | --- |
| IIS security with SQL Server Compact | "IIS Security" in SQL Server Compact Books Online |
| Reporting Services Authentication | [Authentication in Reporting Services](https://learn.microsoft.com/en-us/sql/reporting-services/extensions/security-extension/authentication-in-reporting-services?view=sql-server-ver16) |
| SQL Server Compact and IIS access | "Internet Information Services Security Flowchart" in SQL Server Compact Books Online |

## SQL Server Operating System Files Security

SQL Server uses operating system files for operation and data storage. Best practices for file security requires that you restrict access to these files. The following table contains information about these files.

| **For information about** | **See** |
| --- | --- |
| SQL Server program files | [File Locations for Default and Named Instances of SQL Server](https://learn.microsoft.com/en-us/sql/sql-server/install/file-locations-for-default-and-named-instances-of-sql-server?view=sql-server-ver16) |

SQL Server service packs and upgrades provide enhanced security. To determine the latest available service pack available for SQL Server, see the [SQL Server](https://go.microsoft.com/fwlink/?LinkID=31629) Web site.

You can use the following script to determine the service pack installed on the system.

“SELECT CONVERT(char(20), SERVERPROPERTY('productlevel')); “

## Principals and Database Object Security

Principals are the individuals, groups, and processes granted access to SQL Server. "Securables" are the server, database, and objects the database contains. Each has a set of permissions that can be configured to help reduce the SQL Server surface area. The following table contains information about principals and securables.

| **For information about** | **See** |
| --- | --- |
| Server and database users, roles, and processes | [Principals (Database Engine)](https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/principals-database-engine?view=sql-server-ver16) |
| Server and database objects security | [Securables](https://learn.microsoft.com/en-us/sql/relational-databases/security/securables?view=sql-server-ver16) |
| The SQL Server security hierarchy | [Permissions Hierarchy (Database Engine)](https://learn.microsoft.com/en-us/sql/relational-databases/security/permissions-hierarchy-database-engine?view=sql-server-ver16) |

## Encryption and Certificates

Encryption does not solve access control problems. However, it enhances security by limiting data loss even in the rare occurrence that access controls are bypassed. For example, if the database host computer is misconfigured and a malicious user obtains sensitive data, such as credit card numbers, that stolen information might be useless if it is encrypted. The following table contains more information about encryption in SQL Server.

| **For information about** | **See** |
| --- | --- |
| The encryption hierarchy in SQL Server | [Encryption Hierarchy](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/encryption-hierarchy?view=sql-server-ver16) |
| Implementing secure connections | [Enable Encrypted Connections to the Database Engine (SQL Server Configuration Manager)](https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/enable-encrypted-connections-to-the-database-engine?view=sql-server-ver16) |
| Encryption functions | [Cryptographic Functions (Transact-SQL)](https://learn.microsoft.com/en-us/sql/t-sql/functions/cryptographic-functions-transact-sql?view=sql-server-ver16) |

Certificates are software "keys" shared between two servers that enable secure communications by way of strong authentication. You can create and use certificates in SQL Server to enhance object and connection security. The following table contains information about how to use certificates with SQL Server.

| **For information about** | **See** |
| --- | --- |
| Creating a certificate for use by SQL Server | [CREATE CERTIFICATE (Transact-SQL)](https://learn.microsoft.com/en-us/sql/t-sql/statements/create-certificate-transact-sql?view=sql-server-ver16) |
| Using a certificate with database mirroring | [Use Certificates for a Database Mirroring Endpoint (Transact-SQL)](https://learn.microsoft.com/en-us/sql/database-engine/database-mirroring/use-certificates-for-a-database-mirroring-endpoint-transact-sql?view=sql-server-ver16) |

## Application Security

### Client programs

SQL Server security best practices include writing secure client applications. For more information about how to help secure client applications at the networking layer, see [Client Network Configuration](https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/client-network-configuration?view=sql-server-ver16).

## Windows Defender Application Control (WDAC)

Windows Defender Application Control (WDAC) prevents unauthorized code execution. WDAC is effective way to mitigate the threat of executable file-based malware. For more information, see to [Windows Defender Application Control](https://learn.microsoft.com/en-us/windows/security/threat-protection/windows-defender-application-control/windows-defender-application-control) documentation.

## SQL Server Security Tools, Utilities, Views, and Functions

SQL Server provides tools, utilities, views, and functions that can be used to configure and administer security.

## SQL Server Security Tools and Utilities

The following table contains information about SQL Server tools and utilities that you can use to configure and administer security.

| **For information about** | **See** |
| --- | --- |
| Connecting to, configuring, and controlling SQL Server | [Use SQL Server Management Studio](https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16) |
| Connecting to SQL Server and running queries at the command prompt | [sqlcmd Utility](https://learn.microsoft.com/en-us/sql/tools/sqlcmd-utility?view=sql-server-ver16) |
| Network configuration and control for SQL Server | [SQL Server Configuration Manager](https://learn.microsoft.com/en-us/sql/relational-databases/sql-server-configuration-manager?view=sql-server-ver16) |
| Enabling and disabling features by using Policy-Based Management | [Administer Servers by Using Policy-Based Management](https://learn.microsoft.com/en-us/sql/relational-databases/policy-based-management/administer-servers-by-using-policy-based-management?view=sql-server-ver16) |
| Manipulating symmetric keys for a report server | [rskeymgmt Utility (SSRS)](https://learn.microsoft.com/en-us/sql/reporting-services/tools/rskeymgmt-utility-ssrs?view=sql-server-ver16) |

## SQL Server Security Catalog Views and Functions

The Database Engine exposes security information in several views and functions that are optimized for performance and utility. The following table contains information about security views and functions.

| **For information about** | **See** |
| --- | --- |
| SQL Server security catalog views, which return information about database-level and server-level permissions, principals, roles, and so on. In addition, there are catalog views that provide information about encryption keys, certificates, and credentials. | [Security Catalog Views (Transact-SQL)](https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/security-catalog-views-transact-sql?view=sql-server-ver16) |
| SQL Server security functions, which return information about the current user, permissions and schemas. | [Security Functions (Transact-SQL)](https://learn.microsoft.com/en-us/sql/t-sql/functions/security-functions-transact-sql?view=sql-server-ver16) |
| SQL Server security dynamic management views. | [Security-Related Dynamic Management Views and Functions (Transact-SQL)](https://learn.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/security-related-dynamic-management-views-and-functions-transact-sql?view=sql-server-ver16) |

## Reference

<https://learn.microsoft.com/en-us/sql/relational-databases/security/securing-sql-server?view=sql-server-ver16#application-security>

# SQL Server security best practices

**Applies to:**  SQL Server (all supported versions)  Azure SQL Database  Azure SQL Managed Instance

This article provides information about best practices and guidelines that help establish security for SQL Server. For a comprehensive review of SQL Server security features, see [Securing SQL Server](https://learn.microsoft.com/en-us/sql/relational-databases/security/securing-sql-server?view=sql-server-ver16).

For specific product security best practices, see [Azure SQL Database and SQL Managed Instance](https://learn.microsoft.com/en-us/azure/azure-sql/database/security-best-practice) and [SQL Server on Azure VMs](https://learn.microsoft.com/en-us/azure/azure-sql/virtual-machines/windows/security-considerations-best-practices).

## Overview

A layered security methodology provides a defense-in-depth solution by leveraging multiple security capabilities targeted at different security scopes. The security features made available in SQL Server 2016, and improved in subsequent releases, help counter security threats and provide well-secured database applications.

Azure complies with several industry regulations and standards that can enable you to build a compliant solution with SQL Server running in a virtual machine. For information about regulatory compliance with Azure, see [Azure Trust Center](https://azure.microsoft.com/support/trust-center/).

## Column-level protection

Organizations often need to protect data at the column level as data regarding customers, employees, trade secrets, product data, healthcare, financial, and other sensitive data is often stored in SQL Server databases. Sensitive columns often include national identification/social security numbers, mobile phone numbers, first name, family name, financial account identification, and any other data that could be deemed personally identifiable information (PII).

The methods and features mentioned in this section raise the level of protection at the column level with minimal overhead, and without requiring extensive changes to application code.

Use [Always Encrypted](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-database-engine?view=sql-server-ver16) to encrypt data at rest and over the wire. Encrypted data is only decrypted by client libraries at the application client level. Use [randomized encryption over deterministic](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-database-engine?view=sql-server-ver16#selecting--deterministic-or-randomized-encryption) where possible. [Always Encrypted (with enclaves)](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-enclaves?view=sql-server-ver16) can improve performance for comparison operations such as [BETWEEN, IN, LIKE, DISTINCT, Joins, and more](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-enclaves?view=sql-server-ver16#confidential-queries) for randomized encryption scenarios.

Use [Dynamic Data Masking (DDM)](https://learn.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-ver16#creating-a-dynamic-data-mask) to obfuscate data at the column level when Always Encrypted is not an available option. Dynamic Data Masking (DDM) is [not compatible with Always Encrypted](https://learn.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-ver16#limitations-and-restrictions). Leverage Always Encrypted over dynamic data masking whenever possible.

You can also [GRANT permissions](https://learn.microsoft.com/en-us/sql/t-sql/statements/grant-object-permissions-transact-sql?view=sql-server-ver16) at the column level to a table, view, or table-valued function. Consider the following: - Only SELECT, REFERENCES, and UPDATE permissions can be granted on a column.  
- A table-level DENY does not take precedence over a column-level GRANT.

## Row-level protection

[Row-Level Security (RLS)](https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver16#Typical) enables the ability to leverage user execution context in order to control access to rows in a database table. RLS ensures that users can only see the record that pertains to them. This gives your application 'record level' security without having to make significant changes to your application.

The business logic is encapsulated within table-valued functions controlled by a security policy that toggles the RLS functionality on and off. The security policy also controls the FILTER and BLOCK predicates that are bound to the tables RLS operates against. Use Row-Level Security (RLS) to limit the records that are returned to the user making the call. Leverage [SESSION\_CONTEXT (T-SQL)](https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver16#MidTier) for users who connect to the database through a middle-tier application where application users share the same SQL Server user account. For optimal performance and manageability, follow the [Row-Level Security best practices](https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver16#Best).

**Tip**

Use Row-Level Security (RLS) together with either Always Encrypted or Dynamic Data Masking (DDM) to maximize the security posture of your organization.

## File encryption

[Transparent Data Encryption (TDE)](https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption?view=sql-server-ver16#enable-tde) protects the data at the file level by providing encryption-at-rest to the database files. Transparent Data Encryption (TDE) ensures that database files, backup files, and tempdb files can't be attached and read without proper certificates decrypting database files. Without Transparent Data Encryption (TDE), it is possible for an attacker to take the physical media (drives or backup tapes) and restore or attach the database to read the contents. Transparent Data Encryption (TDE) is supported to work with all other security capabilities in SQL Server. Transparent Data Encryption (TDE) provides real-time I/O encryption and decryption of the data and log files. TDE encryption leverages a database encryption key (DEK) is stored in the user database. The database encryption key can also be protected using a certificate, which is protected by the database master key of the master database.

Use TDE to protect data at rest, backups, and tempdb.

## Auditing and reporting

To [audit SQL Server](https://learn.microsoft.com/en-us/sql/relational-databases/security/auditing/sql-server-audit-database-engine?view=sql-server-ver16), create an audit policy at either the server or database level. Server policies apply to all existing and newly created databases on the server. For simplicity, enable server-level auditing and allow the database-level auditing to inherit the server-level property for all databases.

Audit [tables and columns](https://learn.microsoft.com/en-us/sql/relational-databases/security/auditing/sql-server-audit-database-engine?view=sql-server-ver16) with sensitive data that have security measures applied to them. If a table or column is important enough to need protection by a security capability, then it should be considered important enough to audit. It is especially important to audit and regularly review tables that contain sensitive information but where it is not possible to apply desired security measures due to some kind of application or architectural limitation.

## Identities and authentication

SQL Server supports two [authentication modes](https://learn.microsoft.com/en-us/sql/relational-databases/security/choose-an-authentication-mode?view=sql-server-ver16), Windows authentication mode and 'SQL Server and Windows Authentication mode' (mixed mode).

Logins are separate from database users. First, map logins or Windows groups to database users or roles separately. Next, grant permissions to users, [server roles](https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/server-level-roles?view=sql-server-ver16), and/or [database roles](https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/database-level-roles?view=sql-server-ver16) to access database objects.

SQL Server supports the following types of logins:

* A local Windows user account or Active Directory domain account - SQL Server relies on Windows to authenticate the Windows user accounts.
* Windows group - Granting access to a Windows group grants access to all Windows user logins that are members of the group. Removing a user from a group removes the rights from the user that came from the group. Group membership is the preferred strategy.
* SQL Server login - SQL Server stores the username and a hash of the password in the master database.
* [Contained database users](https://learn.microsoft.com/en-us/sql/relational-databases/security/contained-database-users-making-your-database-portable?view=sql-server-ver16) authenticate SQL Server connections at the database level. A contained database is a database that is isolated from other databases and from the instance of SQL Server (and the master database) that hosts the database. SQL Server supports contained database users for both Windows and SQL Server authentication.

The following recommendations and best practices help secure your identities and authentication methods:

* Use [least-privilege role-based security](https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/getting-started-with-database-engine-permissions?view=sql-server-ver16#grant-the-least-permission) strategies to improve security management.
  + It's standard to place Active Directory users in AD groups, AD groups should exist in SQL Server roles, and SQL Server roles should be granted the minimum permissions required by the application.
* In Azure, leverage least-privilege security by using role-based access (RBAC) controls
* Choose Active Directory over SQL Server authentication whenever possible, and especially choose Active Directory over storing the security at the application or database level.
  + If a user leaves the company it is easy to disable the account.
  + It is also easy to remove users from groups when users change roles or leave the organization. Group security is considered a best practice.
* Leverage [Multi-Factor Authentication](https://learn.microsoft.com/en-us/azure/active-directory/authentication/concept-mfa-howitworks) for accounts that have machine-level access, including accounts that use RDP to log into the machine. This helps guard against credential theft or leaks, as single-factor password-based authentication is a weaker form of authentication with credentials at risk of being compromised or mistakenly given away.
* Require [strong and complex passwords](https://learn.microsoft.com/en-us/sql/relational-databases/security/strong-passwords?view=sql-server-ver16) that cannot be easily guessed, and are not used for any other accounts or purposes. Regularly update passwords and enforce Active Directory policies.
* [Group-Managed Service Accounts (gMSA)](https://learn.microsoft.com/en-us/windows-server/security/group-managed-service-accounts/group-managed-service-accounts-overview) provide automatic password management, simplified service principal name (SPN) management and delegate the management to other administrators.
  + With gMSA, the Windows operating system manages passwords for the account instead of relying on the administrator to manage the password.
  + gMSA automatically updates the account passwords without restarting services.
  + gMSA reduces the administrative surface-level and improves the separation of duties.
* Minimize the rights granted to the AD account of the DBA; Consider a separation of duties that limit access to the virtual machine, the ability to log into the operating system, the ability to modify error and auditing logs, and the ability to install applications and/or features.
* Consider removing DBA accounts from the sysadmin role and granting [CONTROL SERVER](https://learn.microsoft.com/en-us/sql/relational-databases/security/permissions-database-engine?view=sql-server-ver16#chart-of-sql-server-permissions) to DBA accounts rather than making them a member of the sysadmin role. The system admin role does not respect DENY while [CONTROL SERVER](https://learn.microsoft.com/en-us/sql/relational-databases/security/permissions-database-engine?view=sql-server-ver16#chart-of-sql-server-permissions) does.

## Data lineage and data integrity

Keeping historical records of data changes over time can be beneficial to address accidental changes to the data. It can also be useful for application-change auditing and can provide the ability to recover data elements when a bad actor has introduced data changes that were not authorized.

* Leverage [temporal tables](https://learn.microsoft.com/en-us/sql/relational-databases/tables/temporal-tables?view=sql-server-ver16) to preserve record versions over time, and to see data as it has been over the record's life span to provide a historical view of your application's data.
* Temporal Tables can be used to supply a version of the current table at any point in time.

## Security assessment tools and evaluation

The configuration and assessment tools below provide an ability to address surface-area security, identify data security opportunities, and provide a best practice assessment of the security of your SQL Server environment at the instance level.

* [Surface Area Configuration](https://learn.microsoft.com/en-us/sql/relational-databases/security/surface-area-configuration?view=sql-server-ver16) - It is recommended to enable only the features that are required by your environment in order to minimize the number of features that can be attacked by a malicious user.
* [Vulnerability assessment for SQL Server (SSMS)](https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-vulnerability-assessment?view=sql-server-ver16) - SQL vulnerability assessment is a tool in [SSMS v17.4+](https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16) that helps discover, track, and remediate potential database vulnerabilities. The vulnerability assessment is a valuable tool to improve your database security and is executed at the database level, per database.
* [SQL Data Discovery and Classification (SSMS)](https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-data-discovery-and-classification?view=sql-server-ver16) - It is common for DBAs to manage servers and databases and not be aware of sensitivity of the data that is contained in the database. Data Discovery & Classification adds the capability to discover, classify, label and report on the sensitivity level of your data. Data Discovery & Classification is supported starting with [SSMS 17.5](https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16).

## Reference

<https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-server-security-best-practices?view=sql-server-ver16>