Microsoft SQL Server

Microsoft

Keeping your data secure with SQL Server 2016

Technical White Paper

Published: August 2016

Applies to: Microsoft SQL Server 2016 and SQL Server 2014

Summary: This white paper examines data security features, including new and existing SQL Server security features, Windows Server features, and recommended practices, organized by protection layer (client, network, database, and host) and by feature/practice type (access control, data encryption, and proactive monitoring).

Copyright

Contents

Introduction

Microsoft SQL Server has been rated as the most secure database management system (DBMS) for several years. According to the National Institute of Standards and Technology (NIST) public security board, SQL Server has the lowest number of security vulnerabilities across major database vendors. In addition, the Information Technology Industry Council (ITIC)[1] has deemed SQL Server "the most secure database."

SQL Server is designed to be secure by reducing potential security risks by default, allowing administrators to selectively choose connectivity and remote features based on their organization's needs. Security needs are constantly changing, as the landscape of data, volume, accessibility, and potential threats evolves. The latest features in SQL Server 2016 significantly raise the bar for data security, providing new and enhanced features to protect data from various threats.

Understanding security features and best practices can be challenging, even for developers and administrators familiar with previous versions of SQL Server. This paper provides insight on how to frame layers of data protection, the focus areas for security features, and potential scenarios in which features can be applied to limit exposure to security threats.

Security feature areas overview

SQL Server 2016 security features can be organized into three areas of focus. Each focus area represents a data security concept: an approach to securing data from potential breaches or threats. These areas include access control, in which permissions are strictly defined for users and applications to prevent breaches; data encryption, designed to prevent use of sensitive data by unauthorized parties; and proactive monitoring, which focuses on tracking logon attempts and potentially malicious activity.

[1] Information Technology Intelligence Corp. (ITIC), SQL Server Delivers Industry-Leading Security, September 2012.

In addition to new and updated SQL Server security features available in SQL Server 2016, we'll review a few security practices associated with database security, such as Active Directory authentication, Windows Firewall, and hard drive encryption. The intent is to provide insight into capabilities available to developers and database administrators for securing data.

Protection layers overview

In an effort to organize security features by effective scope, we break down the overall concept of data security into several layers. Each layer represents a logical partition of the security landscape by type of access, accessible content, and security concerns.

Security features and practices are grouped into the protection layers to which they apply. Some of these are applicable to more than one layer. In this paper, we will briefly examine each layer outlined above, and review a scenario to illustrate how specific security features and practices can be employed to secure sensitive data.

Data application protection

Data application protection generally focuses on permission: enabling data access for authorized users, and disabling access for others. It's also an area that, from a DBA perspective, is often difficult to control as it's up to developers to provide secure functionality within an application. Traditionally, the logic of obfuscating sensitive data has been developed in the application layer. This requires security to be implemented across all modules and applications accessing the data. Alternatively, special views must be created to avoid exposure of sensitive data. These views can impact database performance and are susceptible to errors. SQL Server 2016 provides features that limit access to data fields containing

sensitive information and includes an additional layer of application security to keep valuable personal data protected even when the data is in-motion.

Consider a scenario where a hospital has stringent requirements to control access to patient data at a more granular level than what simply granting, revoking, or denying permissions provides. For example, a database application that provides access to patient data may require individual doctors to be restricted to accessing information related to only their patients. Similar requirements exist in many environments, including finance, law, government, and military applications.

Row-Level Security (RLS) is a SQL Server feature that simplifies and centralizes access logic in a security policy at a per-row level. RLS enables customers to control access to rows in a database table based on the characteristics of the user executing a T-SQL statement (such as group membership or execution context). RLS uses filter and block predicates to determine which records are visible for a particular transaction, protecting data from unauthorized reads or writes (update/delete). Since predicates are evaluated at the Database Engine, no additional logic is required in the client application.

In this same hospital scenario, consider another example where customer representatives need to address patient questions. When a patient record comes up, the representative needs the ability to see certain information to respond accurately. Some information, such as specific private health details or financial or insurance information, needs to remain confidential for HIPAA compliance. SQL Server 2016 introduces a new feature called Dynamic Data Masking (DDM). With DDM, IT administrators can take simple steps to define policies, or rules, to mask any personally identifiable information that is not needed for the customer interaction. The ability to view "unmasked" data is limited to those with the correct authorization, managed by security group membership. This way, the service representative can view a patient record without having access to confidential information. Patient information is secured, but at the same time, the representative is able to answer questions by accessing appropriate data without compromising privacy.

At the database level, applications and general users should not have access to ad hoc query capabilities, particularly when dealing with sensitive data. A good practice is to prevent user access to data tables, and provide role-based, task-specific functionality with views and stored procedures. Applications should execute queries programmatically as parameterized queries to prevent potential injection attacks. This complements the use of RLS and DDM, as the pre-defined actions of user views and procedures reduce the likelihood of circumventing filtering or obfuscation by the Database Engine.

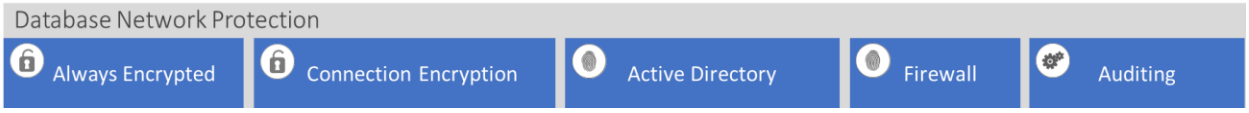
SQL Server can also protect sensitive patient data, such as credit card numbers or national identification numbers (such as US Social Security numbers), whether that data is in-motion or at-rest. New with SQL Server 2016, Always Encrypted allows clients to encrypt sensitive data at the client application level. Encryption keys are externally managed and never revealed to the Database Engine (SQL Database or SQL Server). As a result, Always Encrypted provides a separation between those who own the data (and can

view it) and those who manage the data (but should have no access). This allows the hospital to encrypt data at rest and reduces access to sensitive data by non-authorized personnel, such as DBAs. This also means fewer security clearance requirements for their DBA staff.

Data application layer protection primarily focuses on access control, only providing access to intended parties, and protecting data from all others. With Always Encrypted—coupled with access controls such as Dynamic Data Masking, Row-Level Security, and parameterized queries—developers and administrators can provide additional assurances to clients and customers that data is protected and secure.

Data network protection

Data network protection focuses on secure communication between the database and clients. In addition to SQL Server security features, Windows offers security features that manage access control, which strongly complement SQL Server data encryption and proactive monitoring. Windows Firewall settings enable administrators to determine conditions for which a connection to the server instance is allowed. Windows authentication in SQL Server provides centralized access control with Active Directory. SSL/TLS secures connections to SQL Server. SQL Server auditing logs access attempts and can be further extended to audit additional activities.



Consider an example of an airport hub where the hub needs constant connectivity to the data in-motion and at-rest to ensure safety of passengers and aircraft operations. Data network protection here is imperative since you want specific users to be able to access logs and audit sensitive data while protecting data from threats. Since the data will be in constant flux due to the multitude of airplanes traveling every day, the network and data it produces needs to have strong connectivity and verified authentication.

SQL Server 2016 provides various data network protection features that focus on connectivity (authentication, firewall) and protecting data in-motion. Network administrators can configure Windows Firewall to allow secure SQL Server access. In this scenario, system administrators are able to specify subnets, IP ranges, or specific IP addresses to have access to a specific TCP port, and only for Database Engine access.

Database administrators can employ Windows authentication using identities in Active Directory. Active Directory enables the use of the Kerberos security protocol, as well as additional password policies not available with SQL Server authentication, and also enables token-based authentication between clients and data. With Active Directory authentication, organizations can centrally manage the identities of database users and other Microsoft services. Central ID management provides a single place to manage database users and simplifies permission management. This feature provides an alternative to SQL Server authentication and helps stop the proliferation of user identities across database servers owned by an organization.

IT administrators can also enable encrypted connections for an instance of the SQL Server Database Engine by specifying a certificate using SQL Server Configuration Manager. The server computer must have a certificate provisioned, and the client machine must be set up to trust the certificate's root authority. SQL Server 2016 supports TLS 1.2, the latest cryptographic protocol, for endpoint communication security.

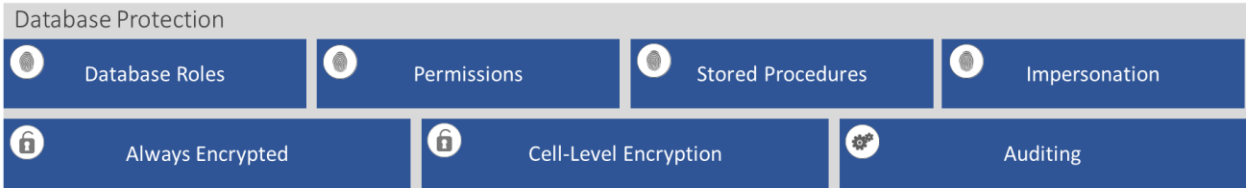
Using Always Encrypted, data is protected both at-rest and in-motion. This protects the data from rogue administrators, backup thieves, and man-in-the-middle attacks. An Always Encrypted-enabled driver installed on the client computer achieves this by automatically encrypting and decrypting sensitive data in the SQL Server client application. The driver encrypts the data in sensitive columns before passing the data to SQL Server, and automatically rewrites queries so that the semantics to the application are preserved. Similarly, the driver transparently decrypts data stored in encrypted database columns and contained in query results.

Auditing an instance of the SQL Server Database Engine or an individual database provides tracking and logging events of the failed connection attempts that occur on the Database Engine. SQL Server 2016 audit lets IT administrators create server audits, which can contain server audit specifications for server-level events, and database audit specifications for database-level events. Audited events can be written to the event logs or to audit files—or raising alerts as needed.

Data network protections ensure the security of data by way of AD authentication, firewall settings, encrypted communications, and proactive monitoring through auditing. SQL Server 2016 supports the latest TLS encryption protocol, and works hand in hand with the centralized identity management of Active Directory. Server audits can alert administrators of potential attempts to breach database security, allowing them to deal with threats in a timely manner. Brought together, developers and administrators can provide solutions with the latest in secure data communications.

Database protection

Database protection relies heavily on separation of roles, and the permissions that are associated with these roles. Roles and permissions not only determine which data is accessible, but the actions a user is allowed to take in relation to the data, whether a user can view unencrypted sensitive data, or make modifications to the database schema itself. Secure role management focuses on providing the minimum level of permission required for a role to perform its tasks. Further protection is possible by using impersonation to temporarily allow elevated permissions on-demand.



Consider a wholesaler providing bulk products to retailers across the region. Employees have different needs when it comes to accessing operational data. For example, account managers maintain customer

contact data, process payments, and log orders, while those in fulfillment review orders, update status, and manage inventory. Both departments need to run periodic reports on their operations. Customers need access to view invoices, order status, and history. Their interactions with the data are largely managed through role assignment as members of Active Directory groups (for example, "Customers," "AccountManagers," and "FulfillmentTeam" mapped to specific roles in SQL Server).

In this scenario, client users do not have access to tables, nor the ability to execute custom or ad hoc queries. Instead, views or stored procedures are made accessible to roles, ensuring only those operations intended for use within a specific role are available for use. For example, fulfillment team users can query a view that displays all open orders by target date, while customers can access stored procedures that return their most recent order information.

IT services has roles for interacting with production data as well. DBAs monitor the overall system, but do not require the ability to create or view specific orders. A DBA can have access to a set of modules (functions, procedures, queues, and triggers) that can impersonate a "Customers" or "AccountManagers" role to troubleshoot performance. Impersonation within a module allows temporary elevations within the module. These modules would grant temporary access to data or processes, but do not require full ad hoc query capabilities. Additionally, temporary elevation of permissions can be used via Windows Active Directory Just Enough Administration (JEA) to elevate permissions in SQL Server in addition to Windows. Active Directory and JEA are further discussed at the database-host protection level.

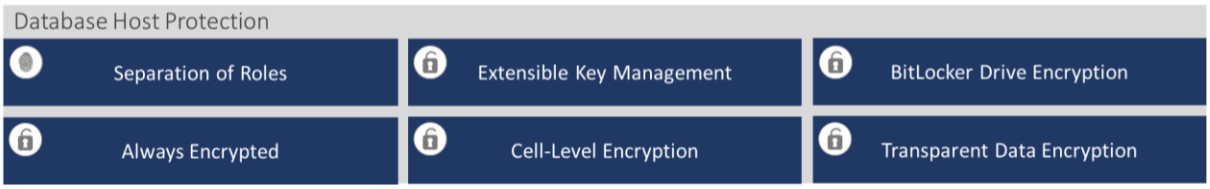
DBA-specific stored procedures enable impersonating a role temporarily to assist in identifying causes of performance degradation. With Always Encrypted or CLE, sensitive data such as customer contact or payment information is encrypted, allowing detailed troubleshooting to be conducted without compromising data security.

Audit administrators maintain a set of audits flagging unusual activities. The audit admin needs the ability to create and manage audits and triggers. The auditor creates triggers to alert the team to issues for various reasons, such as order totals that fall outside of a specific range, stagnated open orders, or signs of malicious DBA activity.

Database protection relies on features and practices from all feature areas: access control, data encryption, and proactive monitoring. Separation of roles and deliberate permissions management are key in protecting data from a rouge administrator while providing functionality to users. Providing users with bare-minimum permission also protects them from attack. Impersonation grants temporary, monitored access for administrators to troubleshoot operations when needed, without granting full-time permissions. Stored procedures, when implemented as secure entry points for prescribed actions, allow developers to provide targeted functionality without risking security. Auditing gives administrators early insight to potential security risks. In concert, these features and practices serve to raise the bar on database security to a new level.

Database host protection

Database host protection focuses on protecting data from those with access to the host machine/operating system, or physical access to the hardware. Database host protection relies heavily on encryption, as well as separation of roles. SQL Server 2016 offers several encryption features, some of which can be stacked to greatly reduce the risk of a breach. In addition to SQL Server-specific features, host-level features such as Bitlocker Drive Encryption, which protects the physical drive, and Extensible Key Management, which provides off-host (or even off-site) encryption key storage, further enhance the safeguarding of sensitive data. Additionally, a strong policy of role separation at the host with JEA and Database Engine access provides another level of protection.



Our next scenario is of a research and development department with a vast store of proprietary design and test data that pertains to a potential government contract. For numerous reasons, research data must remain on-site, and protected from unauthorized access. This includes system administrators who have or need elevated access to systems, and potential theft of physical hard drives.

Always Encrypted applies to this scenario as well, as data is stored in its encrypted state. Since the client applications using this data handle the decryption process, and encryption keys are stored separate from the Database Engine, Always Encrypted is able to protect data both in-motion and at-rest, including any potentially malicious acts from database administrators.

In addition to using Always Encrypted for end-to-end data encryption, TDE is employed to protect on-disk data as well as backups. With EKM, encryption keys stored in individual smart cards enable DBAs to backup/restore, or otherwise manage data as needed. System or OS administrators are able to log on to systems to perform maintenance and management tasks, but do not have the ability to decrypt database files. This protects data from potential malicious actions from a rouge administrator.

To further reduce the risk of breaches, Windows Server introduces Just Enough Administration (JEA). The concept behind JEA is for administrators to spend the majority of their time in standard logons, without elevated permissions. JEA will temporarily provide an authorized user with elevated permissions to perform a series of tasks that require elevation. This action is heavily logged and monitored, further protecting systems from malicious activity. JEA is a Windows PowerShell toolkit that can be used to apply temporary elevation to SQL Server instances and Windows Server administrative functions. JEA is supported on Windows Server 2008 R2 and above, and is built into Windows Server 2016.

Related to JEA is Privileged Access Management (PAM), which further protects user groups that control access to computers on a domain. Similar to JEA, PAM provides more oversight and control over the activities of privileged administrators. PAM separates privileged accounts to a separate (bastion), trusted domain. Accounts in the bastion domain are created through an approval process, and have time-limited

memberships. This allows users to request special access as-needed with a high degree of accountability, and without the need for permanent elevated permissions.

Bitlocker Drive Encryption adds an additional layer of protection. With Bitlocker enabled, entire volumes on disk are encrypted, with keys stored on the server's TPM or attached HSM. Should a hard drive be physically removed and connected to another computer in an attempt to steal information, encrypted volumes on the drive remain encrypted, unable to be decrypted with the absence of the original TPM/HSM key store.

Host protection extends beyond the Database Engine instance to the host environment and hardware. Data at rest is protected at multiple levels with encryption; the keys for which can be maintained in an external key store. Database files and backups are encrypted transparently. Separation of roles separates administrators from data read/write capabilities, and JEA further enhances this distinction. Bitlocker Drive Encryption adds yet another layer of encryption, protecting files of data even in the event of a physical breach. By extending beyond the Database Engine, administrators can tightly lock down data and keep it secure.

Conclusion

SQL Server is designed to be secure by design, reducing potential security risks by default. Security needs are constantly changing, as the landscape of data, volume, accessibility, and potential threats evolves. SQL Server 2016 provides features that significantly raise the bar for data security, giving developers and administrators the means to protect data from various threats.

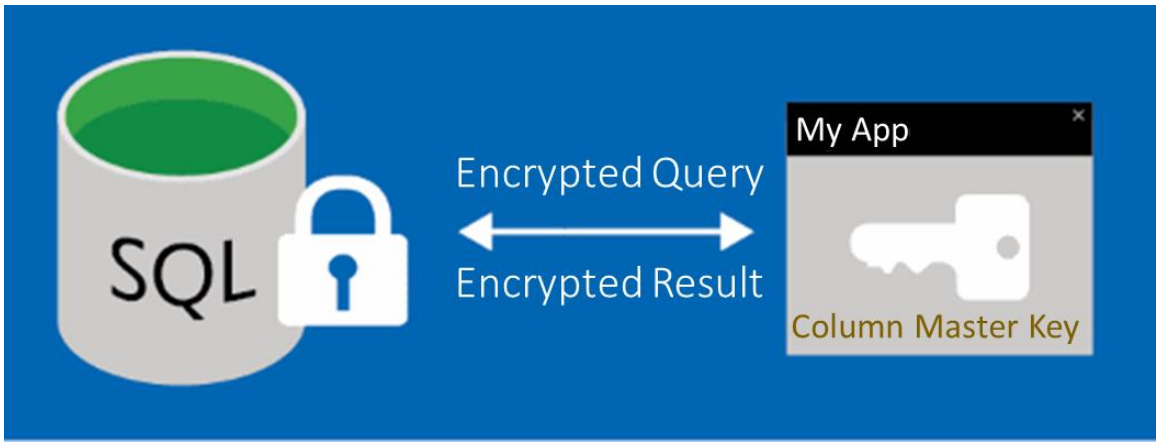
In this paper, SQL Server and Windows Server security features and best practices have been organized in a way to assist developers and administrators to more easily understand data security. With this framework of protection layers and feature areas, organizations are able to deploy more secure solutions, protecting data from end to end while instilling confidence in users, clients, and customers about the security of their critically sensitive information.

Appendix: Security features in detail

Data encryption

Always Encrypted (AE)

Always Encrypted is available in SQL Server 2016 as a means to protect data in-motion and at-rest. AE provides added security at several protection levels. With AE, sensitive data is encrypted via ADO.NET before the query is sent across the network. For inserts and updates, encrypted data is stored without decryption. Decryption occurs on authorized clients with the appropriate encryption key, ensuring that sensitive data is obfuscated to any user other than the intended clients. From an application protection perspective, AE not only protects data in-motion, but provides the added assurance that without the correct encryption key, even if data access is compromised, sensitive data is rendered useless.



From a network protection layer perspective, AE adds to the existing security of encrypted communications by encrypting sensitive data at the client prior to transmission over the network, and decrypting at the client after receiving data. This means that should information be transmitted in an insecure manner, or if network security is compromised, sensitive data is still protected by the master encryption key.

From a database protection layer perspective, data is encrypted at all stages. Since encryption/decryption is handled at the client level, sensitive data is treated and stored as binary data at the server, without knowledge or access to the column encryption key.

See: https://msdn.microsoft.com/en-us/library/mt163865.aspx

Cell-Level Encryption

Cell-Level Encryption (CLE) is a method of encrypting data at-rest. CLE differs from TDE in that while TDE encrypts and decrypts data transparently at the I/O layer, CLE encryption/decryption occurs explicitly at query time. CLE offers more granular control, allowing developers to define specific columns to encrypt. CLE also allows developers to select from several encryption methods: by passphrase, certificate, or asymmetric or symmetric key. CLE works with database permissions and role separation to distinguish which users can access decrypted data from specific columns. Encryption/decryption happens at the server level, so while data remains encrypted in the buffer pool, it is transmitted/received in clear text.

See: https://msdn.microsoft.com/en-us/library/ms179331.aspx

Extensible Key Management

SQL Server Extensible Key Management (EKM) enables the encryption keys that protect the database files to be stored in an off-box device such as a smartcard, USB device, or EKM/HSM module. This protects database files from being accessed by system administrators or other users with elevated OS permissions. EKM provides an additional authorization check, further supporting separation of duties, as well as support for hardware-based, high-performance encryption and decryption.

See: https://msdn.microsoft.com/en-us/library/bb895340.aspx

Transparent Data Encryption

Transparent Data Encryption (TDE) is data protected at rest, particularly with EKM or key store, where OS administrators aren't able to decrypt directly. TDE allows organizations to encrypt data when it is stored on a disk and decrypt it when it is read into memory. This enables software developers to encrypt database files, log files, and backup files without changing existing applications.

TDE has been updated to take advantage of the latest Intel AES-NI hardware acceleration found in most Intel processors over the last several years. This results in significantly faster performance. Following initial encryption, there is little CPU impact for many workloads.

TDE now also supports storage of memory-optimized OLTP tables. This allows for greater security, along with the performance enhancements provided by memory optimization.

See: https://msdn.microsoft.com/en-us/library/bb934049.aspx

Bitlocker Drive Encryption

Bitlocker Drive Encryption encrypts drive volumes to protect data from being accessed by those who have physical access to the disk. With Bitlocker, the encryption keys used to protect data are stored in the Trusted Platform Module (TPM) of the host machine, or Hardware Security Module (HSM) connected to the host machine. In the event of a drive being connected to another machine, none of data on the encrypted volume is accessible. Without the proper encryption key, the data is completely locked.

See: http://support.microsoft.com/kb/2855131

Connection Encrypting

Secure communications are provided with SSL/TLS, or more specifically TLS 1.2 with SQL Server 2016. TLS created private connections using symmetric cryptography to encrypt data transmissions. TLS is effective in protecting data from attacks through the use of endpoint authentication with a third-party certificate store, shared-secret negotiation, and message integrity checks.

See: https://support.microsoft.com/kb/3135244

Access control

Row-Level Security

Row-Level Security (RLS) ensures that data is made available to users who should have access, while protecting data from those who should not have access. Access restriction occurs at the Database Engine level, as opposed to the application tier, simplifying the design and coding security of applications.

RLS functions through the applications of filter and block predicates. Filter predicates restrict the rows available for read operations (SELECT, UPDATE, DELETE), filtering results behind the scenes. Block predicates prevent write operations (AFTER INSERT, AFTER UPDATE, BEFORE UPDATE, BEFORE DELETE) on rows that do not match the predicate.

Row-level access is enforced by a security policy, which invokes an inline table-valued function as a security predicate. With filter predicates, filtered rows are excluded silently, with no indication of any filtering having occurred. With block predicates, operations that violate the predicate will fail with an error.

Implementing RLS requires forethought and planning, as well as following best practices.

See: https://msdn.microsoft.com/en-us/library/dn765131.aspx#Anchor_4

Dynamic Data Masking

Dynamic Data Masking (DDM) hides sensitive data by obfuscating all or some of the value with a data mask. The underlying data for a masked column is unchanged, but when queried, non-privileged users see the masked version of the data as opposed to the actual values. This can be particularly useful when users are working with information of a sensitive nature such as Social Security numbers or financial

account numbers. DDM provides several built-in data mask functions: default, email, custom string, and random. The *default* function fully masks fields based upon the data type. *Email* exposes the first letter of an email address, and a '.com' suffix. *Custom string* allows developers to create custom masks according to their needs. *Random* provides a random masking function that replaces a value with a random number within a specified range.

See: https://msdn.microsoft.com/en-us/library/mt130841.aspx

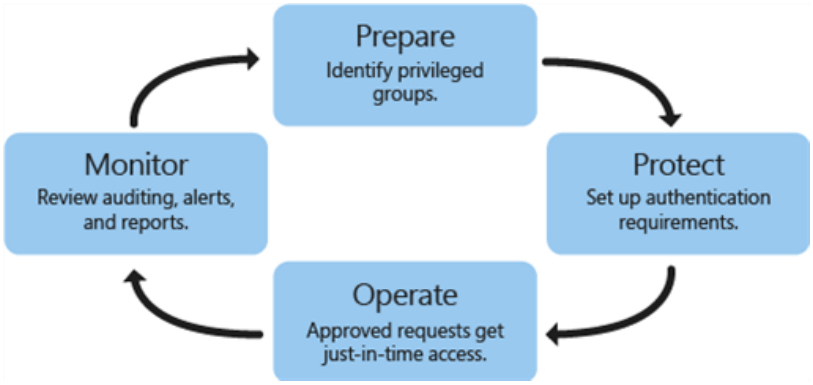
Just Enough Administration

Just Enough Administration (JEA) is a security feature for Windows Server designed to provide the minimum number of permissions to system administrators until elevation permissions are needed. JEA will temporarily provide an authorized user with elevated permissions to perform a series of tasks that require elevation, under heavy monitoring and control. This greatly reduces the risk of a rouge administrator or other malicious user being able to attack a system internally. JEA is a Windows PowerShell toolkit that is supported on Windows Server 2008 R2 and above, and is built into Windows Server 2016.

See: https://msdn.microsoft.com/en-us/library/dn896648.aspx

Privileged Access Management

Privileged Access Management (PAM) is based on Microsoft Identity Manager (MIM) and builds on the concept of just-in-time administration. PAM operates by use of a second, trusted, bastion Active Directory (AD) forest. Privileged access is granted upon request from the bastion forest with a time limit. This access request is approved based on configured MIM policies. The entire process is monitored and logged, enabling alerts, auditing, and reports of privileged access.



See: https://docs.microsoft.com/en-us/microsoft-identity-manager/pam/privileged-identity-management-for-active-directory-domain-services

Active Directory authentication

Active Directory (AD) authentication provides several important security features for data protection. Centralized identity management with AD enables an alternative to SQL Server authentication, which can be cumbersome when dealing with keeping permissions and passwords up-to-date. AD enables the use

of the Kerberos security protocol, as well as additional password policies not available with SQL Server authentication, and also enables token-based authentication between clients and data. In combination with JEA, AD authentication significantly improves support for least privilege and reduces risk against insider attacks.

Windows Firewall

Windows Firewall acts to prevent unauthorized machine access by limiting the activity on communication ports (both TCP and UDP) to a select set of pre-approved ports and/or applications. By default, Windows Firewall closes port 1433 (the default TCP/IP port for SQL Server default instances), preventing network access to the Database Engine. Administrators can enable this port, open a specific port if another port is used, or add sqlserver.exe as an exception to blocked programs listed in Windows Firewall—which allows for the use of dynamic ports. Each of these options poses potential security risks.

See: https://msdn.microsoft.com/en-us/library/ms144228.aspx

Separation of roles

Separation of roles is an important concept in security, which serves to protect data from being accessed without authorization by those who would traditionally have had unrestricted access to resources and data. This operates on the basis of providing the minimum amount of access needed for a given role to perform its function, and nothing more. For example, a DBA would require access to database engines and have the ability to manage database access permissions and schema changes, but does not require access to sensitive personal or financial data contained within operational databases. System administration is an activity separate from database administration. Users with domain/OS administrative privileges should not have unrestricted access to the database schema or data.

Permissions

Permissions are assigned to principals such as users, logons, or roles. Permissions have a hierarchy, based on the scope of influence. SQL Server-level permissions determine actions at the Database Engine level, while database permissions pertain to a specific database. Best practices are to assign permissions to the roles that need them, as opposed to specific users. Logons should be assigned to the role with the appropriate permissions to perform day-to-day tasks. Logons can loosely correlate with Windows or Active Directory groups, where groups somewhat define a job function, which can then map to SQL Server logons and, by extension, roles. Windows or Active Directory users are then organized into groups, linking individual users to the roles and permissions required. It is possible to assign an AD user to a SQL Server logon, but doing so requires specific permissions management for that individual and poses a potential hole in secure management.

See: https://msdn.microsoft.com/en-us/library/mt667986.aspx

Stored procedures and views

Stored procedures offer a convenient method of performing otherwise complicated tasks, but from a security perspective, it allows developers to encapsulate tasks to ensure they are always executed in a

prescribed manner. Using stored procedures and views offers the benefit of restricting access to the underlying tables and ad hoc querying, as well as the ability to prescribe specific actions users can execute. Stored procedures can be defined to accept/require parameters and return scalar values or data sets. Views are read-only, predefined queries, and can be optimized for performance for client experience.

Impersonation

Impersonation is done through the EXECUTE AS clause. This allows selection of a specific context under which a command is run. This is particularly useful when certain operations are outside the bounds of regular use for a particular group of users. Impersonation can be used in any T-SQL execution statement, providing programmatic or as-needed permissions to perform a task without providing permissions manually or on a long-term basis.

See: https://msdn.microsoft.com/en-us/library/ms181362.aspx

Extending database impersonation by using EXECUTE AS:
https://msdn.microsoft.com/en-us/library/ms188304(v=sql.105).aspx

Using EXECUTE AS in modules:
https://msdn.microsoft.com/en-us/library/ms178106(v=sql.105).aspx

Parameterized queries

Parameterized queries is not a specific feature for SQL Server, but is a solid best practice for security. Using parameterized queries protects the query from injection attacks. Similar to stored procedures, query parameters are strongly typed and abstracted from the SQL statement, packaged within the SqlCommand Object. As a general guideline, stored procedures are preferred in terms of maintainability and reuse, but parameterized queries are a valid approach to protecting queries at an application level.

Proactive monitoring

Auditing

Auditing involves tracking and logging events that occur at a database or SQL Server level. Built-in auditing tracks logon attempts (particularly failed attempts) to log suspicious connection attempts. SQL Server allows for the use of extended events in creating an audit, enabling administrators to create custom audits for various events to help track down and avert malicious activity. Additional triggers can be defined to raise events such as unusual time periods or duplicate logons from multiple systems that may indicate a breach.

SQL Server allows developers to define events to write custom information to an audit log. This is particularly useful in tracking potentially malicious actions by authenticated users. Triggers may include extremely large transactions, unusually frequent transfers in a short time for commerce or banking scenarios, or triggers on timing of requests such as a series of transactions occurring outside of standard business hours.

More information

The following websites offer more information about topics discussed in this white paper:

- Security Center for SQL Server Database Engine and Azure SQL Database
  - https://msdn.microsoft.com/en-us/library/bb510589.aspx
- Just Enough Administration: Windows PowerShell security controls help protect enterprise data
  - https://msdn.microsoft.com/en-us/library/dn896648.aspx
- Privileged Access Management for Active Directory Domain Services
  - https://docs.microsoft.com/en-us/microsoft-identity-manager/pam/privileged-identity-management-for-active-directory-domain-services
- BitLocker overview
  - https://technet.microsoft.com/en-us/library/hh831713(v=ws.11).aspx

Feedback

Did this paper help you? Please give us your feedback by telling us on a scale of 1 (poor) to 5 (excellent) how you rate this paper and why you have given it this rating. More specifically:

- Are you rating it highly because of relevant examples, helpful screen shots, clear writing, or another reason?
- Are you rating it poorly because of examples that don't apply to your concerns, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

Please send your feedback to: sqlsrvwpfeedback@microsoft.com