

CIS Microsoft SQL Server 2022 Benchmark

v1.3.0 - 05-29-2026

Terms of Use

Please see the below link for our current terms of use:

<https://www.cisecurity.org/cis-securesuite/cis-securesuite-membership-terms-of-use/>

For information on referencing and/or citing CIS Benchmarks in 3rd party documentation (including using portions of Benchmark Recommendations) please contact CIS Legal (legalnotices@cisecurity.org) and request guidance on copyright usage.

NOTE: It is **NEVER** acceptable to host a CIS Benchmark in **ANY** format (PDF, etc.) on a 3rd party (non-CIS owned) site.

Table of Contents

Terms of Use	1
Table of Contents	2
Overview	5
Important Usage Information	5
Key Stakeholders	5
Apply the Correct Version of a Benchmark	6
Exceptions	6
Remediation	7
Summary	7
Target Technology Details	8
Intended Audience.....	8
Consensus Guidance	9
Typographical Conventions.....	10
Recommendation Definitions.....	11
Title.....	11
Assessment Status.....	11
Automated	11
Manual.....	11
Profile	11
Description.....	11
Rationale Statement	11
Impact Statement.....	12
Audit Procedure.....	12
Remediation Procedure.....	12
Default Value.....	12
References	12
CIS Critical Security Controls® (CIS Controls®).....	12
Additional Information.....	12
Profile Definitions	13
Acknowledgements	14
Recommendations	15
1 Installation, Updates and Patches	15
1.1 Ensure Latest SQL Server Cumulative and Security Updates are Installed (Manual)	16
1.2 Ensure Single-Function Member Servers are Used (Manual).....	18
2 Surface Area Reduction	20

2.1 Ensure 'Ad Hoc Distributed Queries' Server Configuration Option is set to '0' (Automated)	21
2.2 Ensure 'CLR Enabled' Server Configuration Option is set to '0' (Automated)	23
2.3 Ensure 'Cross DB Ownership Chaining' Server Configuration Option is set to '0' (Automated)	26
2.4 Ensure 'Database Mail XPs' Server Configuration Option is set to '0' (Automated)	28
2.5 Ensure 'Ole Automation Procedures' Server Configuration Option is set to '0' (Automated)	30
2.6 Ensure 'Remote Access' Server Configuration Option is set to '0' (Automated)	32
2.7 Ensure 'Remote Admin Connections' Server Configuration Option is set to '0' (Automated)	34
2.8 Ensure 'Scan For Startup Procs' Server Configuration Option is set to '0' (Automated)	36
2.9 Ensure 'Trustworthy' Database Property is set to 'Off' (Automated)	38
2.10 Ensure Unnecessary SQL Server Protocols are set to 'Disabled' (Manual)	40
2.11 Ensure SQL Server is configured to use non-standard ports (Automated)	42
2.12 Ensure 'Hide Instance' option is set to 'Yes' for Production SQL Server instances (Automated)	45
2.13 Ensure the 'sa' Login Account is set to 'Disabled' (Automated)	48
2.14 Ensure the 'sa' Login Account has been renamed (Automated)	50
2.15 Ensure 'AUTO_CLOSE' is set to 'OFF' on contained databases (Automated)	52
2.16 Ensure no login exists with the name 'sa' (Automated)	54
2.17 Ensure 'clr strict security' Server Configuration Option is set to '1' (Automated)	56
3 Authentication and Authorization	58
3.1 Ensure 'Server Authentication' Property is set to 'Windows Authentication Mode' (Automated)	59
3.2 Ensure CONNECT permissions on the 'guest' user is Revoked within all SQL Server databases (Automated)	61
3.3 Ensure 'Orphaned Users' are Dropped From SQL Server Databases (Automated)	63
3.4 Ensure SQL Authentication is not used in contained databases (Automated)	65
3.5 Ensure the SQL Server's MSSQL Service Account is Not an Administrator (Manual)	67
3.6 Ensure the SQL Server's SQLAgent Service Account is Not an Administrator (Manual)	69
3.7 Ensure the SQL Server's Full-Text Service Account is Not an Administrator (Manual)	71
3.8 Ensure only the default permissions specified by Microsoft are granted to the public server role (Automated)	73
3.9 Ensure Windows BUILTIN groups are not SQL Logins (Automated)	75
3.10 Ensure Windows local groups are not SQL Logins (Automated)	77
3.11 Ensure the public role in the msdb database is not granted access to SQL Agent proxies (Automated)	79
3.12 Ensure the 'SYSADMIN' Role is Limited to Administrative or Built-in Accounts (Manual)	81
3.13 Ensure no admin role membership in MSDB database (Automated)	83
3.14 Ensure 'Control Server' permission is not granted (Manual)	85
4 Password Policies	88
4.1 Ensure 'MUST_CHANGE' Option is set to 'ON' for All SQL Authenticated Logins (Manual)	89
4.2 Ensure 'CHECK_EXPIRATION' Option is set to 'ON' for All SQL Authenticated Logins Within the Sysadmin Role (Automated)	91
4.3 Ensure 'CHECK_POLICY' Option is set to 'ON' for All SQL Authenticated Logins (Automated)	94
5 Auditing and Logging	96
5.1 Ensure 'Maximum number of error log files' is set to greater than or equal to '12' (Automated)	97
5.2 Ensure 'Default Trace Enabled' Server Configuration Option is set to '1' (Automated)	100
5.3 Ensure 'Login Auditing' is set to 'failed logins' (Automated)	102

5.4 Ensure 'SQL Server Audit' is set to capture both 'failed' and 'successful logins' (Automated)	104
6 Application Development.....	109
6.1 Ensure Database and Application User Input is Sanitized (Manual)	110
6.2 Ensure 'CLR Assembly Permission Set' is set to 'SAFE_ACCESS' for All CLR Assemblies (Automated)	112
7 Encryption.....	114
7.1 Ensure 'Symmetric Key encryption algorithm' is set to 'AES_128' or higher in non-system databases (Automated).....	115
7.2 Ensure Asymmetric Key Size is set to 'greater than or equal to 2048' in non-system databases (Automated).....	117
7.3 Ensure Database Backups are Encrypted (Automated).....	119
7.4 Ensure Network Encryption is Configured and Enabled (Automated).....	121
7.5 Ensure Databases are Encrypted with TDE (Automated)	123
8 Appendix: Additional Considerations	125
8.1 Ensure 'SQL Server Browser Service' is configured correctly (Manual)	126
9 Appendix - Establishing an Audit/Scan User.....	128
<i>Appendix: Summary Table</i>	<i>129</i>
<i>Appendix: Change History</i>	<i>133</i>

Overview

All CIS Benchmarks™ (Benchmarks) focus on technical configuration settings used to maintain and/or increase the security of the addressed technology, and they should be used in **conjunction** with other essential cyber hygiene tasks like:

- Monitoring the base operating system and applications for vulnerabilities and quickly updating with the latest security patches.
- End-point protection (Antivirus software, Endpoint Detection and Response (EDR), etc.).
- Logging and monitoring user and system activity.

In the end, the Benchmarks are designed to be a key **component** of a comprehensive cybersecurity program.

Important Usage Information

All Benchmarks are available free for non-commercial use from the [CIS Website](#). They can be used to manually assess and remediate systems and applications. In lieu of manual assessment and remediation, there are several tools available to assist with assessment:

- [CIS Configuration Assessment Tool \(CIS-CAT® Pro Assessor\)](#)
- [CIS Benchmarks™ Certified 3rd Party Tooling](#)

These tools make the hardening process much more scalable for large numbers of systems and applications.

NOTE: Some tooling focuses only on the Benchmark Recommendations that can be fully automated (skipping ones marked **Manual**). It is important that **ALL** Recommendations (**Automated** and **Manual**) be addressed since all are important for properly securing systems and are typically in scope for audits.

Key Stakeholders

Cybersecurity is a collaborative effort, and cross functional cooperation is imperative within an organization to discuss, test, and deploy Benchmarks in an effective and efficient way. The Benchmarks are developed to be best practice configuration guidelines applicable to a wide range of use cases. In some organizations, exceptions to specific Recommendations will be needed, and this team should work to prioritize the problematic Recommendations based on several factors like risk, time, cost, and labor. These exceptions should be properly categorized and documented for auditing purposes.

Apply the Correct Version of a Benchmark

Benchmarks are developed and tested for a specific set of products and versions and applying an incorrect Benchmark to a system can cause the resulting pass/fail score to be incorrect. This is due to the assessment of settings that do not apply to the target systems. To assure the correct Benchmark is being assessed:

- **Deploy the Benchmark applicable to the way settings are managed in the environment:** An example of this is the Microsoft Windows family of Benchmarks, which have separate Benchmarks for Group Policy, Intune, and Stand-alone systems based upon how system management is deployed. Applying the wrong Benchmark in this case will give invalid results.
- **Use the most recent version of a Benchmark:** This is true for all Benchmarks, but especially true for cloud technologies. Cloud technologies change frequently and using an older version of a Benchmark may have invalid methods for auditing and remediation.

Exceptions

The guidance items in the Benchmarks are called recommendations and not requirements, and exceptions to some of them are expected and acceptable. The Benchmarks strive to be a secure baseline, or starting point, for a specific technology, with known issues identified during Benchmark development are documented in the Impact section of each Recommendation. In addition, organizational, system specific requirements, or local site policy may require changes as well, or an exception to a Recommendation or group of Recommendations (e.g. A Benchmark could Recommend that a Web server not be installed on the system, but if a system's primary purpose is to function as a Webserver, there should be a documented exception to this Recommendation for that specific server).

In the end, exceptions to some Benchmark Recommendations are common and acceptable, and should be handled as follows:

- The reasons for the exception should be reviewed cross-functionally and be well documented for audit purposes.
- A plan should be developed for mitigating, or eliminating, the exception in the future, if applicable.
- If the organization decides to accept the risk of this exception (not work toward mitigation or elimination), this should be documented for audit purposes.

It is the responsibility of the organization to determine their overall security policy, and which settings are applicable to their unique needs based on the overall risk profile for the organization.

Remediation

CIS has developed [Build Kits](#) for many technologies to assist in the automation of hardening systems. Build Kits are designed to correspond to Benchmark's "Remediation" section, which provides the manual remediation steps necessary to make that Recommendation compliant to the Benchmark.

When remediating systems (changing configuration settings on deployed systems as per the Benchmark's Recommendations), please approach this with caution and test thoroughly.

The following is a reasonable remediation approach to follow:

- CIS Build Kits, or internally developed remediation methods should never be applied to production systems without proper testing.
- Proper testing consists of the following:
 - Understand the configuration (including installed applications) of the targeted systems. Various parts of the organization may need different configurations (e.g., software developers vs standard office workers).
 - Read the Impact section of the given Recommendation to help determine if there might be an issue with the targeted systems.
 - Test the configuration changes with representative lab system(s). If issues arise during testing, they can be resolved prior to deploying to any production systems.
 - When testing is complete, initially deploy to a small sub-set of production systems and monitor closely for issues. If there are issues, they can be resolved prior to deploying more broadly.
 - When the initial deployment above is completed successfully, iteratively deploy to additional systems and monitor closely for issues. Repeat this process until the full deployment is complete.

Summary

Using the Benchmarks Certified tools, working as a team with key stakeholders, being selective with exceptions, and being careful with remediation deployment, it is possible to harden large numbers of deployed systems in a cost effective, efficient, and safe manner.

NOTE: As previously stated, the PDF versions of the CIS Benchmarks™ are available for free, non-commercial use on the [CIS Website](#). All other formats of the CIS Benchmarks™ (MS Word, Excel, and [Build Kits](#)) are available for CIS [SecureSuite®](#) members.

CIS-CAT® Pro is also available to CIS [SecureSuite®](#) members.

Target Technology Details

This document provides prescriptive guidance for establishing a secure configuration posture for Microsoft SQL Server 2022. This guide was tested against Microsoft SQL Server 2022. To obtain the latest version of this guide, please visit <http://benchmarks.cisecurity.org>. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

Intended Audience

This benchmark is intended for system and application administrators, security specialists, auditors, help desk, and platform deployment personnel who plan to develop, deploy, assess, or secure solutions that incorporate Microsoft SQL Server 2022 on a Microsoft Windows platform.

Consensus Guidance

This CIS Benchmark™ was created using a consensus review process comprised of a global community of subject matter experts. The process combines real world experience with data-based information to create technology specific guidance to assist users to secure their environments. Consensus participants provide perspective from a diverse set of backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

Each CIS Benchmark undergoes two phases of consensus review. The first phase occurs during initial Benchmark development. During this phase, subject matter experts convene to discuss, create, and test working drafts of the Benchmark. This discussion occurs until consensus has been reached on Benchmark recommendations. The second phase begins after the Benchmark has been published. During this phase, all feedback provided by the Internet community is reviewed by the consensus team for incorporation in the Benchmark. If you are interested in participating in the consensus process, please visit <https://workbench.cisecurity.org/>.

Typographical Conventions

The following typographical conventions are used throughout this guide:

Convention	Meaning
<code>Stylized Monospace font</code>	Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented.
<code>Monospace font</code>	Used for inline code, commands, UI/Menu selections or examples. Text should be interpreted exactly as presented.
<code><Monospace font in brackets></code>	Text set in angle brackets denote a variable requiring substitution for a real value.
<i>Italic font</i>	Used to reference other relevant settings, CIS Benchmarks and/or Benchmark Communities. Also, used to denote the title of a book, article, or other publication.
Bold font	Additional information or caveats things like Notes , Warnings , or Cautions (usually just the word itself and the rest of the text normal).

Recommendation Definitions

The following defines the various components included in a CIS recommendation as applicable. If any of the components are not applicable it will be noted, or the component will not be included in the recommendation.

Title

Concise description for the recommendation's intended configuration.

Assessment Status

An assessment status is included for every recommendation. The assessment status indicates whether the given recommendation can be automated or requires manual steps to implement. Both statuses are equally important and are determined and supported as defined below:

Automated

Represents recommendations for which assessment of a technical control can be fully automated and validated to a pass/fail state. Recommendations will include the necessary information to implement automation.

Manual

Represents recommendations for which assessment of a technical control cannot be fully automated and requires all or some manual steps to validate that the configured state is set as expected. The expected state can vary depending on the environment.

Profile

A collection of recommendations for securing a technology or a supporting platform. Most benchmarks include at least a Level 1 and Level 2 Profile. Level 2 extends Level 1 recommendations and is not a standalone profile. The Profile Definitions section in the benchmark provides the definitions as they pertain to the recommendations included for the technology.

Description

Detailed information pertaining to the setting with which the recommendation is concerned. In some cases, the description will include the recommended value.

Rationale Statement

Detailed reasoning for the recommendation to provide the user a clear and concise understanding on the importance of the recommendation.

Impact Statement

Any security, functionality, or operational consequences that can result from following the recommendation.

Audit Procedure

Systematic instructions for determining if the target system complies with the recommendation.

Remediation Procedure

Systematic instructions for applying recommendations to the target system to bring it into compliance according to the recommendation.

Default Value

Default value for the given setting in this recommendation, if known. If not known, either not configured or not defined will be applied.

References

Additional documentation relative to the recommendation.

CIS Critical Security Controls® (CIS Controls®)

The mapping between a recommendation and the CIS Controls is organized by CIS Controls version, Safeguard, and Implementation Group (IG). The Benchmark in its entirety addresses the CIS Controls safeguards of (v7) "5.1 - Establish Secure Configurations" and (v8) "4.1 - Establish and Maintain a Secure Configuration Process" so individual recommendations will not be mapped to these safeguards.

Additional Information

Supplementary information that does not correspond to any other field but may be useful to the user.

Profile Definitions

The following configuration profiles are defined by this Benchmark:

- **Level 1 - Database Engine**

Items in this profile apply to Microsoft SQL Server 2022 and intend to:

- be practical and prudent;
- provide a clear security benefit; and
- not inhibit the utility of the technology beyond acceptable means.

- **Level 1 - AWS RDS**

Items in this profile are applicable to Microsoft SQL Server 2022 on AWS RDS and intend to:

- be practical and prudent;
- provide a clear security benefit; and
- not inhibit the utility of the technology beyond acceptable means.

- **Level 2 - Database Engine**

This profile extends the "Level 1 - Database Engine" profile. Items in this profile apply to Microsoft SQL Server 2022 and exhibit one or more of the following characteristics:

- are intended for environments or use cases where security is paramount;
- acts as defense in depth measure; and
- may impact the utility or performance of the technology.

Acknowledgements

This Benchmark exemplifies the great things a community of users, vendors, and subject matter experts can accomplish through consensus collaboration. The CIS community thanks the entire consensus team with special recognition to the following individuals who contributed greatly to the creation of this guide:

Contributor

Rob Kraft
Emad Al-Mousa
Steinar Andersen
Danny De Haan

Editor

Brian Kelley
Tim Harrison
Sean McCown
Krishna Rayavaram

Recommendations

1 Installation, Updates and Patches

This section contains recommendations related to installing and patching SQL Server.

1.1 Ensure Latest SQL Server Cumulative and Security Updates are Installed (Manual)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

SQL Server patches contain program updates that fix security and product functionality issues found in the software. These patches can be installed with a security update, which is a single patch, or a cumulative update which is a group of patches. The SQL Server version and patch levels should be the most recent compatible with the organizations' operational needs.

Rationale:

Using the most recent SQL Server software, along with all applicable patches can help limit the possibilities for vulnerabilities in the software. The installation version and/or patches applied during setup should be established according to the needs of the organization.

Audit:

To determine your SQL Server patch level, run the following code snippet.

```
SELECT SERVERPROPERTY('ProductLevel') as SP_installed,  
SERVERPROPERTY('ProductVersion') as Version,  
SERVERPROPERTY('ProductUpdateLevel') as 'ProductUpdate_Level',  
SERVERPROPERTY('ProductUpdateReference') as 'KB_Number';
```

Remediation:

Identify the current version and patch level of your SQL Server instances and ensure they contain the latest security fixes. Make sure to test these fixes in your test environments before updating production instances.

The most recent SQL Server patches can be found here:

<https://learn.microsoft.com/en-us/troubleshoot/sql/releases/download-and-install-latest-updates>

Default Value:

Cumulative and security updates are not installed by default.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/install-windows/latest-updates-for-microsoft-sql-server>
2. <https://support.microsoft.com/en-us/help/4041553/sql-server-service-packs-are-discontinued-starting-from-sql-server>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.2 <u>Ensure Authorized Software is Currently Supported</u> Ensure that only currently supported software is designated as authorized in the software inventory for enterprise assets. If software is unsupported, yet necessary for the fulfillment of the enterprise's mission, document an exception detailing mitigating controls and residual risk acceptance. For any unsupported software without an exception documentation, designate as unauthorized. Review the software list to verify software support at least monthly, or more frequently.			
v7	2.2 <u>Ensure Software is Supported by Vendor</u> Ensure that only software applications or operating systems currently supported by the software's vendor are added to the organization's authorized software inventory. Unsupported software should be tagged as unsupported in the inventory system.			

1.2 Ensure Single-Function Member Servers are Used (Manual)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

It is recommended that SQL Server software be installed on a dedicated server. This architectural consideration affords security flexibility in that the database server can be placed on a separate subnet allowing access only from particular hosts and over particular protocols. Degrees of availability are easier to achieve as well - over time, an enterprise can move from a single database server to a failover to a cluster using load balancing or to some combination thereof.

Rationale:

It is easier to manage (i.e. reduce) the attack surface of the server hosting SQL Server software if the only surfaces to consider are the underlying operating system, SQL Server itself, and any security/operational tooling that may additionally be installed. As noted in the description, availability can be more easily addressed if the database is on a dedicated server.

Impact:

It is difficult to see any reasonably adverse impact to making this architectural change, once the costs of making the change have been paid. Custom applications may need to be modified to accommodate database connections over the wire rather than on the host (i.e. using TCP/IP instead of Named Pipes). Additional hardware and operating system licenses may be required to make these architectural changes.

Audit:

Ensure that no other roles are enabled for the underlying operating system and that no excess tooling is installed, per enterprise policy.

Remediation:

Uninstall excess tooling and/or remove unnecessary roles from the underlying operating system.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>3.12 Segment Data Processing and Storage Based on Sensitivity</u> Segment data processing and storage based on the sensitivity of the data. Do not process sensitive data on enterprise assets intended for lower sensitivity data.			
v7	<u>2.10 Physically or Logically Segregate High Risk Applications</u> Physically or logically segregated systems should be used to isolate and run software that is required for business operations but incur higher risk for the organization.			

2 Surface Area Reduction

SQL Server offers various configuration options, some of them can be controlled by the `sp_configure` stored procedure. This section contains the listing of the corresponding recommendations.

2.1 Ensure 'Ad Hoc Distributed Queries' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Enabling Ad Hoc Distributed Queries allows users to query data and execute statements on external data sources. This functionality should be disabled.

Rationale:

This feature can be used to remotely access and exploit vulnerabilities on remote SQL Server instances and to run unsafe Visual Basic for Application functions.

Audit:

Run the following T-SQL command:

```
SELECT name, CAST(value as int) as value_configured, CAST(value_in_use as int) as value_in_use
FROM sys.configurations
WHERE name = 'Ad Hoc Distributed Queries';
```

Both value columns must show 0 to be compliant.

Remediation:

For AWS RDS Instances, please refer to the documentation for using Parameter Groups here:

[Working with parameter groups](#)

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;
RECONFIGURE;
EXECUTE sp_configure 'Ad Hoc Distributed Queries', 0;
RECONFIGURE;
GO
EXECUTE sp_configure 'show advanced options', 0;
RECONFIGURE;
```

Default Value:

By default, this option is disabled (0).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/ad-hoc-distributed-queries-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.4 <u>Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	4.5 <u>Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	9.2 <u>Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.2 Ensure 'CLR Enabled' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **clr enabled** option specifies whether user assemblies can be run by SQL Server.

Rationale:

Enabling use of CLR assemblies widens the attack surface of SQL Server and puts it at risk from both inadvertent and malicious assemblies.

Impact:

If CLR assemblies are in use, applications may need to be rearchitected to eliminate their usage before disabling this setting. Alternatively, some organizations may allow this setting to be enabled **1** for assemblies created with the **SAFE** permission set, but disallow assemblies created with the riskier **UNSAFE** and **EXTERNAL_ACCESS** permission sets. To find user-created assemblies, run the following query in all databases, replacing **<database_name>** with each database name:

```
USE [<database_name>]
GO
SELECT name AS Assembly_Name, permission_set_desc
FROM sys.assemblies
WHERE is_user_defined = 1;
GO
```

Audit:

Run the following T-SQL command:

```
SELECT name,
       CAST(value as int) as value_configured,
       CAST(value_in_use as int) as value_in_use
FROM sys.configurations
WHERE name = 'clr strict security';
```

If both values are **1**, this recommendation is Not Applicable. Otherwise, run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'clr enabled';
```

Both value columns must show 0 to be compliant.

Remediation:

For AWS RDS Instances, please refer to the documentation for using Parameter Groups here: [Working with parameter groups](#)

Run the following T-SQL command:

```
EXECUTE sp_configure 'clr enabled', 0;  
RECONFIGURE;
```

Default Value:

By default, this option is disabled (0).

References:

1. <https://learn.microsoft.com/en-us/sql/t-sql/statements/create-assembly-transact-sql>

Additional Information:

If **clr strict security** is set to 1 this recommendation is not applicable. By default, **clr strict security** is enabled and treats **SAFE** and **EXTERNAL_ACCESS** assemblies as if they were marked **UNSAFE**. Though not recommended, the **clr strict security** option can be disabled for backward compatibility. This recommendation has been retained for environments configured for backwards compatibility.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>16.7 Use Standard Hardening Configuration Templates for Application Infrastructure</u> Use standard, industry-recommended hardening configuration templates for application infrastructure components. This includes underlying servers, databases, and web servers, and applies to cloud containers, Platform as a Service (PaaS) components, and SaaS components. Do not allow in-house developed software to weaken configuration hardening.			
v7	<u>18.11 Use Standard Hardening Configuration Templates for Databases</u> For applications that rely on a database, use standard hardening configuration templates. All systems that are part of critical business processes should also be tested.			

2.3 Ensure 'Cross DB Ownership Chaining' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **cross db ownership chaining** option controls cross-database ownership chaining across all databases at the instance (or server) level.

Rationale:

When enabled, this option allows a member of the **db_owner** role in a database to gain access to objects owned by a login in any other database, causing an unnecessary information disclosure. When required, cross-database ownership chaining should only be enabled for the specific databases requiring it instead of at the instance level for all databases by using the **ALTER DATABASE<database_name>SET DB_CHAINING ON** command. This database option may not be changed on the **master**, **model**, or **tempdb** system databases.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'cross db ownership chaining';
```

Both value columns must show **0** to be compliant.

Remediation:

For AWS RDS Instances, please refer to the documentation for using Parameter Groups here: [Working with parameter groups](#)

Run the following T-SQL command:

```
EXECUTE sp_configure 'cross db ownership chaining', 0;  
RECONFIGURE;  
GO
```

Default Value:

By default, this option is disabled (**0**).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/cross-db-ownership-chaining-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.4 Ensure 'Database Mail XPs' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **Database Mail XPs** option controls the ability to generate and transmit email messages from SQL Server.

Rationale:

Disabling the **Database Mail XPs** option reduces the SQL Server surface, eliminates a DOS attack vector and channel to exfiltrate data from the database server to a remote host.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'Database Mail XPs';
```

Both value columns must show 0 to be compliant.

Remediation:

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'Database Mail XPs', 0;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Default Value:

By default, this option is disabled (0).

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/database-mail/database-mail>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>4.4 Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	<u>4.5 Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	<u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.5 Ensure 'Ole Automation Procedures' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **Ole Automation Procedures** option controls whether OLE Automation objects can be instantiated within Transact-SQL batches. These are extended stored procedures that allow SQL Server users to execute functions external to SQL Server.

Rationale:

Enabling this option will increase the attack surface of SQL Server and allow users to execute functions in the security context of SQL Server.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'Ole Automation Procedures';
```

Both value columns must show **0** to be compliant.

Remediation:

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'Ole Automation Procedures', 0;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Default Value:

By default, this option is disabled (**0**).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/ole-automation-procedures-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 <u>Establish and Maintain a Secure Configuration Process</u> Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 <u>Establish Secure Configurations</u> Maintain documented, standard security configuration standards for all authorized operating systems and software.			

2.6 Ensure 'Remote Access' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **remote access** option controls the execution of local stored procedures on remote servers or remote stored procedures on local server.

Rationale:

Functionality can be abused to launch a Denial-of-Service (DoS) attack on remote servers by off-loading query processing to a target.

Impact:

Per Microsoft: This feature may be removed in a future version of Microsoft SQL Server. Do not use this feature in new development work, and modify applications that currently use this feature as soon as possible. Use **sp_addlinkedserver** instead.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'remote access';
```

Both value columns must show **0** to be compliant.

Remediation:

For AWS RDS Instances, please refer to the documentation for using Parameter Groups here: [Working with parameter groups](#)

Otherwise, run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'remote access', 0;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Restart the Database Engine.

Default Value:

By default, this option is enabled (1).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-the-remote-access-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.4 <u>Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	4.5 <u>Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	9.2 <u>Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.7 Ensure 'Remote Admin Connections' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **remote admin connections** option controls whether a client application on a remote computer can use the Dedicated Administrator Connection (DAC).

Rationale:

The Dedicated Administrator Connection (DAC) lets an administrator access a running server to execute diagnostic functions or Transact-SQL statements, or to troubleshoot problems on the server, even when the server is locked or running in an abnormal state and not responding to a SQL Server Database Engine connection. In a cluster scenario, the administrator may not actually be logged on to the same node that is currently hosting the SQL Server instance and thus is considered "remote". Therefore, this setting should usually be enabled (**1**) for SQL Server failover clusters; otherwise, it should be disabled (**0**) which is the default.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'remote admin connections'  
AND SERVERPROPERTY('IsClustered') = 0;
```

If no data is returned, the instance is a cluster and this recommendation is not applicable. If data is returned, then both the value columns must show **0** to be compliant.

Remediation:

Run the following T-SQL command on non-clustered installations:

```
EXECUTE sp_configure 'remote admin connections', 0;  
RECONFIGURE;  
GO
```

Default Value:

By default, this option is disabled (**0**), only local connections may use the DAC.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/remote-admin-connections-server-configuration-option>

Additional Information:

If it's a clustered installation, this option must be enabled as a clustered SQL Server cannot bind to localhost and DAC will be unavailable otherwise. Enable it for clustered installations. Disable it for standalone installations where not required.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.4 <u>Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	4.5 <u>Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	9.2 <u>Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.8 Ensure 'Scan For Startup Procs' Server Configuration Option is set to '0' (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **scan for startup procs** option, if enabled, causes SQL Server to scan for and automatically run all stored procedures that are set to execute upon service startup.

Rationale:

Enforcing this control reduces the threat of an entity leveraging these facilities for malicious purposes.

Impact:

Setting Scan for Startup Procedures to **0** will prevent certain audit traces and other commonly used monitoring stored procedures from re-starting on start up. Additionally, replication requires this setting to be enabled (**1**) and will automatically change this setting if needed.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'scan for startup procs';
```

Both value columns must show **0** to be compliant.

Remediation:

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'scan for startup procs', 0;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Restart the Database Engine.

Default Value:

By default, this option is disabled (**0**).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-the-scan-for-startup-procs-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 <u>Establish and Maintain a Secure Configuration Process</u> Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 <u>Establish Secure Configurations</u> Maintain documented, standard security configuration standards for all authorized operating systems and software.			

2.9 Ensure 'Trustworthy' Database Property is set to 'Off' (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **TRUSTWORTHY** database option allows database objects to access objects in other databases under certain circumstances.

Rationale:

Provides protection from malicious CLR assemblies or extended procedures.

Audit:

Run the following T-SQL query to list any databases with a Trustworthy database property value of **ON**:

```
SELECT name
FROM sys.databases
WHERE is_trustworthy_on = 1
AND name != 'msdb';
```

No rows should be returned to be compliant.

Remediation:

Execute the following T-SQL statement against the databases (replace **<database_name>** below) returned by the Audit Procedure:

```
ALTER DATABASE [<database_name>] SET TRUSTWORTHY OFF;
```

Default Value:

By default, this database property is **OFF** (**is_trustworthy_on = 0**), except for the **msdb** database in which it is required to be **ON**.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/trustworthy-database-property>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.10 Ensure Unnecessary SQL Server Protocols are set to 'Disabled' (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

SQL Server supports Shared Memory, Named Pipes, and TCP/IP protocols. However, SQL Server should be configured to use the bare minimum required based on the organization's needs.

Rationale:

Using fewer protocols minimizes the attack surface of SQL Server and, in some cases, can protect it from remote attacks.

Impact:

The Database Engine (MSSQL and SQLAgent) services must be stopped and restarted for the change to take effect.

Audit:

Open **SQL Server 2022 Configuration Manager**; go to the **SQL Server Network Configuration**. Ensure that only required protocols are enabled.

Remediation:

Open **SQL Server 2022 Configuration Manager**; go to the **SQL Server Network Configuration**. Ensure that only required protocols are enabled. Disable protocols not necessary.

Default Value:

By default, TCP/IP and Shared Memory protocols are enabled on all commercial editions.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/enable-or-disable-a-server-network-protocol>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>4.4 Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	<u>4.5 Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	<u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.11 Ensure SQL Server is configured to use non-standard ports (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

If installed, a default SQL Server instance will be assigned a default port of **TCP:1433** for TCP/IP communication. Administrators can also manually configure named instances to use **TCP:1433** for communication. **TCP:1433** is a widely known SQL Server port and this port assignment should be changed. In a multi-instance scenario, each instance must be assigned its own dedicated TCP/IP port.

Rationale:

Changing the default SQL Server port from **TCP:1433** reduces exposure to automated reconnaissance and opportunistic attacks that specifically target this well-known port. Most large-scale scanning tools and exploit frameworks focus on default service ports and do not perform exhaustive enumeration, so servers operating on non-standard ports are less likely to appear in mass vulnerability datasets or automated attack campaigns. While this measure does not prevent targeted attacks, it supports defense-in-depth by increasing the time and effort required for adversary discovery and lateral movement.

Impact:

Changing the default port will force the DAC (Dedicated Administrator Connection) to listen on a random port. Also, it might make benign applications, such as application firewalls, require special configuration. In general, you should set a static port for consistent usage by applications, including firewalls, instead of using dynamic ports which will be chosen randomly at each SQL Server start up.

Audit:

Run the following T-SQL:

```
IF (select value_data from sys.dm_server_registry where value_name =  
'ListenOnAllIPs') = 1  
SELECT count(*) FROM sys.dm_server_registry WHERE registry_key like '%IPAll%'  
and value_name like '%Tcp%' and value_data='1433'  
ELSE  
SELECT count(*) FROM sys.dm_server_registry WHERE value_name like '%Tcp%' and  
value_data='1433';
```

A value of **0** is required to be compliant.

Remediation:

1. In **SQL Server Configuration Manager**, in the console pane, expand **SQL Server Network Configuration**, expand Protocols for *<InstanceName>*, and then double-click the TCP/IP protocol
2. In the **TCP/IP Properties** dialog box, on the **IP Addresses** tab, several IP addresses appear in the format **IP1**, **IP2**, up to **IPA11**. One of these is for the IP address of the loopback adapter, **127.0.0.1**. Additional IP addresses appear for each IP Address on the computer.
3. Under **IPA11**, change the **TCP Port** field from **1433** to a non-standard port or leave the **TCP Port** field empty and set the **TCP Dynamic Ports** value to **0** to enable dynamic port assignment and then click **OK**.
4. In the console pane, click **SQL Server Services**.
5. In the details pane, right-click **SQL Server (<InstanceName>)** and then click **Restart**, to stop and restart SQL Server.

Default Value:

By default, default SQL Server instances listen on to TCP/IP traffic on TCP port **1433** and named instances use dynamic ports.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-a-server-to-listen-on-a-specific-tcp-port>

Additional Information:

In the case of AWS RDS, this is only configurable during the build process.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>4.4 Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	<u>4.5 Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	<u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.12 Ensure 'Hide Instance' option is set to 'Yes' for Production SQL Server instances (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

Non-clustered SQL Server instances within production environments should be designated as hidden to prevent advertisement by the SQL Server Browser service.

Rationale:

Designating production SQL Server instances as hidden leads to a more secure installation because they cannot be enumerated. However, clustered instances may break if this option is selected.

Impact:

This method only prevents the instance from being listed on the network. If the instance is hidden (not exposed by SQL Browser), then connections will need to specify the server and port in order to connect. It does not prevent users from connecting to server if they know the instance name and port.

If you hide a clustered named instance, the cluster service may not be able to connect to the SQL Server. Please refer to the Microsoft documentation reference.

Audit:

Perform either the GUI or T-SQL method shown:

GUI Method

1. In **SQL Server 2022 Configuration Manager**, expand **SQL Server Network Configuration**, right-click **Protocols for <InstanceName>**, and then select **Properties**.
2. On the **Flags** tab, in the **Hide Instance** box, if **Yes** is selected, it is compliant.

T-SQL Method

Execute the following T-SQL.

```
DECLARE @getValue INT;
EXEC master.sys.xp_instance_regread
    @rootkey = N'HKEY_LOCAL_MACHINE',
    @key = N'SOFTWARE\Microsoft\Microsoft SQL
Server\MSSQLServer\SuperSocketNetLib',
    @value_name = N'HideInstance',
    @value = @getValue OUTPUT;
SELECT @getValue;
```

A value of **1** should be returned to be compliant.

Remediation:

Perform either the GUI or T-SQL method shown:

GUI Method

1. In **SQL Server 2022 Configuration Manager**, expand **SQL Server Network Configuration**, right-click **Protocols for <InstanceName>**, and then select **Properties**.
2. On the **Flags** tab, in the **Hide Instance** box, select **Yes**, and then click **OK** to close the dialog box. The change takes effect immediately for new connections.

T-SQL Method

Execute the following T-SQL to remediate:

```
EXEC master.sys.xp_instance_regwrite
    @rootkey = N'HKEY_LOCAL_MACHINE',
    @key = N'SOFTWARE\Microsoft\Microsoft SQL
Server\MSSQLServer\SuperSocketNetLib',
    @value_name = N'HideInstance',
    @type = N'REG_DWORD',
    @value = 1;
```

Default Value:

By default, this option is disabled (**0**).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/hide-an-instance-of-sql-server-database-engine>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>4.4 Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	<u>4.5 Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	<u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.13 Ensure the 'sa' Login Account is set to 'Disabled' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **sa** account is a widely known and often widely used SQL Server account with sysadmin privileges. This is the original login created during installation and always has the **principal_id=1** and **sid=0x01**.

Rationale:

Enforcing this control reduces the probability of an attacker executing brute force attacks against a well-known principal.

Impact:

It is not a good security practice to code applications or scripts to use the **sa** account. However, if this has been done, disabling the **sa** account will prevent scripts and applications from authenticating to the database server and executing required tasks or functions.

Audit:

Use the following syntax to determine if the **sa** account is disabled. Checking for **sid=0x01** ensures that the original **sa** account is being checked in case it has been renamed per best practices.

```
SELECT name, is_disabled
FROM sys.server_principals
WHERE sid = 0x01
AND is_disabled = 0;
```

No rows should be returned to be compliant. An **is_disabled** value of **0** indicates the login is currently enabled and therefore needs remediation.

Remediation:

Execute the following T-SQL query:

```
USE [master]
GO
DECLARE @tsql nvarchar(max)
SET @tsql = 'ALTER LOGIN ' + SUSER_NAME(0x01) + ' DISABLE'
EXEC (@tsql)
GO
```

Default Value:

By default, the **sa** login account is disabled at install time when Windows Authentication Mode is selected. If mixed mode (SQL Server and Windows Authentication) is selected at install, the default for the **sa** login is enabled.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-server-principals-transact-sql>
2. <https://learn.microsoft.com/en-us/sql/t-sql/statements/alter-login-transact-sql>
3. <https://learn.microsoft.com/en-us/sql/relational-databases/security/choose-an-authentication-mode>

Additional Information:

In the case of AWS RDS the default name for this account is **rdsa** instead of **sa**.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.3 Disable Dormant Accounts Delete or disable any dormant accounts after a period of 45 days of inactivity, where supported.			
v7	16.8 Disable Any Unassociated Accounts Disable any account that cannot be associated with a business process or business owner.			

2.14 Ensure the 'sa' Login Account has been renamed (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **sa** account is a widely known and often widely used SQL Server login with sysadmin privileges. The **sa** login is the original login created during installation and always has **principal_id=1** and **sid=0x01**.

Rationale:

It is more difficult to launch password-guessing and brute-force attacks against the **sa** login if the name is not known.

Impact:

It is not a good security practice to code applications or scripts to use the **sa** login. However, if this has been done, renaming the **sa** login will prevent scripts and applications from authenticating to the database server and executing required tasks or functions.

Audit:

Use the following syntax to determine if the **sa** login (principal) is renamed.

```
SELECT name
FROM sys.server_principals
WHERE sid = 0x01;
```

A name of **sa** indicates the account has not been renamed and therefore needs remediation.

Remediation:

Replace the **<different_user>** value within the below syntax and execute to rename the **sa** login.

```
ALTER LOGIN sa WITH NAME = <different_user>;
```

Default Value:

By default, the **sa** login name is 'sa'.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/choose-an-authentication-mode>

Additional Information:

In the case of AWS RDS the default name for this account is **rd**sa instead of **sa**.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 <u>Establish and Maintain a Secure Configuration Process</u> Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 <u>Establish Secure Configurations</u> Maintain documented, standard security configuration standards for all authorized operating systems and software.			

2.15 Ensure 'AUTO_CLOSE' is set to 'OFF' on contained databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

AUTO_CLOSE determines if a given database is closed or not after a connection terminates. If enabled, subsequent connections to the given database will require the database to be reopened and relevant procedure caches to be rebuilt.

Rationale:

Because authentication of users for contained databases occurs within the database not at the server\instance level, the database must be opened every time to authenticate a user. The frequent opening/closing of the database consumes additional server resources and may contribute to a denial of service.

Audit:

Perform the following to find contained databases that are not configured as prescribed:

```
SELECT name, containment, containment_desc, is_auto_close_on
FROM sys.databases
WHERE containment <> 0 and is_auto_close_on = 1;
```

No rows should be returned to be compliant.

Remediation:

Execute the following T-SQL, replacing *<database_name>* with each database name found by the Audit Procedure:

```
ALTER DATABASE <database_name> SET AUTO_CLOSE OFF;
```

Default Value:

By default, the database property **AUTO_CLOSE** is **OFF** which is equivalent to **is_auto_close_on = 0**.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/databases/security-best-practices-with-contained-databases>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 <u>Establish and Maintain a Secure Configuration Process</u> Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 <u>Establish Secure Configurations</u> Maintain documented, standard security configuration standards for all authorized operating systems and software.			

2.16 Ensure no login exists with the name 'sa' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **sa** login (e.g. principal) is a widely known and often widely used SQL Server account. Therefore, there should not be a login called **sa** even when the original **sa** login (**principal_id** = 1) has been renamed.

Rationale:

Enforcing this control reduces the probability of an attacker executing brute force attacks against a well-known principal name.

Impact:

It is not a good security practice to code applications or scripts to use the **sa** account. Given that it is a best practice to rename and disable the **sa** account, some 3rd party applications check for the existence of a login named **sa** and if it doesn't exist, creates one. Removing the **sa** login will prevent these scripts and applications from authenticating to the database server and executing required tasks or functions.

Audit:

Use the following syntax to determine if there is an account named **sa**.

```
SELECT principal_id, name
FROM sys.server_principals
WHERE name = 'sa';
```

No rows should be returned to be compliant.

Remediation:

Execute the appropriate **ALTER** statement below based on the **principal_id** returned for the login named **sa**. Replace the **<different_name>** value within the below syntax and execute to rename the **sa** login.

```
USE [master]
GO
-- If principal_id = 1 or the login owns database objects, rename the sa
login
ALTER LOGIN [sa] WITH NAME = <different_name>;
GO
```

Default Value:

The login with **principal_id = 1** is named **sa** by default.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 <u>Establish and Maintain a Secure Configuration Process</u> Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 <u>Establish Secure Configurations</u> Maintain documented, standard security configuration standards for all authorized operating systems and software.			

2.17 Ensure 'clr strict security' Server Configuration Option is set to '1' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The **clr strict security** option specifies whether the engine applies the **PERMISSION_SET** on the assemblies.

Rationale:

Enabling use of CLR assemblies widens the attack surface of SQL Server and puts it at risk from both inadvertent and malicious assemblies.

Impact:

If CLR assemblies are in use, applications may need to be rearchitected to eliminate their usage before enabling this setting. To find user-created assemblies, run the following query in all databases, replacing **<database_name>** with each database name:

```
USE [<database_name>]
GO
SELECT name AS Assembly_Name, permission_set_desc
FROM sys.assemblies
WHERE is_user_defined = 1;
GO
```

Audit:

Run the following T-SQL command:

```
SELECT name,
       CAST(value as int) as value_configured,
       CAST(value_in_use as int) as value_in_use
FROM sys.configurations
WHERE name = 'clr strict security';
```

Both value columns must show **1** to be compliant.

Remediation:

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'clr strict security', 1;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Default Value:

By default, this option is Enabled (1).

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/clr-strict-security>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.7 <u>Use Standard Hardening Configuration Templates for Application Infrastructure</u> Use standard, industry-recommended hardening configuration templates for application infrastructure components. This includes underlying servers, databases, and web servers, and applies to cloud containers, Platform as a Service (PaaS) components, and SaaS components. Do not allow in-house developed software to weaken configuration hardening.			
v8	16.8 <u>Separate Production and Non-Production Systems</u> Maintain separate environments for production and non-production systems.			
v7	18.9 <u>Separate Production and Non-Production Systems</u> Maintain separate environments for production and nonproduction systems. Developers should not have unmonitored access to production environments.			
v7	18.11 <u>Use Standard Hardening Configuration Templates for Databases</u> For applications that rely on a database, use standard hardening configuration templates. All systems that are part of critical business processes should also be tested.			

3 Authentication and Authorization

This section contains recommendations related to SQL Server's authentication and authorization mechanisms.

3.1 Ensure 'Server Authentication' Property is set to 'Windows Authentication Mode' (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

Use **Windows Authentication** to validate attempted connections.

Rationale:

Windows provides a more robust authentication mechanism than SQL Server authentication.

Impact:

Changing the login mode configuration requires a restart of the service.

Audit:

Execute the following syntax:

```
SELECT SERVERPROPERTY('IsIntegratedSecurityOnly') as [login_mode];
```

A **login_mode** of **1** indicates the **Server Authentication** property is set to **Windows Authentication Mode**. A **login_mode** of **0** indicates mixed mode authentication. A value of **1** is required to be compliant.

Remediation:

Perform either the GUI or T-SQL method shown:

GUI Method

1. Open **SQL Server Management Studio**.
2. Open the **Object Explorer** tab and connect to the target SQL Server instance.
3. Right click the instance name and select **Properties**.
4. Select the **Security** page from the left menu.
5. Set the **Server authentication** setting to **Windows Authentication Mode**.

T-SQL Method

Run the following T-SQL in a Query Window:

```
USE [master]
GO
EXEC xp_instance_regwrite N'HKEY_LOCAL_MACHINE',
N'Software\Microsoft\MSSQLServer\MSSQLServer', N'LoginMode', REG_DWORD, 1
GO
```

Restart the SQL Server service for the change to take effect.

Default Value:

Windows Authentication Mode

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/server-properties-security-page>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.6 <u>Centralize Account Management</u> Centralize account management through a directory or identity service.			
v7	16.2 <u>Configure Centralized Point of Authentication</u> Configure access for all accounts through as few centralized points of authentication as possible, including network, security, and cloud systems.			

3.2 Ensure CONNECT permissions on the 'guest' user is Revoked within all SQL Server databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Remove the right of the **guest** user to connect to SQL Server databases, except for **master**, **msdb**, **tempdb**, and, on AWS RDS instances, **rdsadmin**.

Rationale:

A login assumes the identity of the **guest** user when a login has access to SQL Server but does not have access to a database through its own account and the database has a **guest** user account. Revoking the **CONNECT** permission for the **guest** user will ensure that a login is not able to access database information without explicit access to do so.

Impact:

When **CONNECT** permission to the **guest** user is revoked, a SQL Server instance login must be mapped to a database user explicitly in order to have access to the database.

Audit:

Run the following code snippet for each database (replacing **<database_name>** as appropriate) in the instance to determine if the **guest** user has **CONNECT** permission.

```
USE <database_name>;
GO
SELECT DB_NAME() AS DatabaseName, 'guest' AS Database_User,
[permission_name], [state_desc]
FROM sys.database_permissions
WHERE [grantee_principal_id] = DATABASE_PRINCIPAL_ID('guest')
AND [state_desc] LIKE 'GRANT%'
AND [permission_name] = 'CONNECT'
AND DB_NAME() NOT IN ('master', 'tempdb', 'msdb');
```

No rows should be returned. On AWS RDS instance, if only **rdsadmin** is returned, this is considered a pass and therefore compliant.

Remediation:

The following code snippet revokes **CONNECT** permissions from the **guest** user in a database. Replace **<database_name>** as appropriate:

```
USE <database_name>;
GO
REVOKE CONNECT FROM guest;
```

Default Value:

The **guest** user cannot have the **CONNECT** permission revoked in **master**, **msdb**, **tempdb**, and, on AWS RDS instances, **rdsadmin**; however, this permission should be revoked in all other databases on the SQL Server instance.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/policy-based-management/guest-permissions-on-user-databases>

Additional Information:

The **guest** user cannot have the **CONNECT** permission revoked in **master**, **msdb** and **tempdb**, but this permission should be revoked in all other databases on the SQL Server instance.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

3.3 Ensure 'Orphaned Users' are Dropped From SQL Server Databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

A database user for which the corresponding SQL Server login is undefined or is incorrectly defined on a server instance cannot log in to the instance and is referred to as orphaned and should be removed.

Rationale:

Orphan users should be removed to avoid potential misuse of those broken users in any way.

Audit:

Run the following T-SQL query in each database to identify orphan users.

```
USE <database_name>;
GO

SELECT *
FROM sys.database_principals DP
WHERE DP.sid NOT IN (
    SELECT L.sid
    FROM sys.syslogins L
    WHERE L.sid = DP.sid
)
AND (SELECT containment FROM sys.databases WHERE name = DB_NAME()) = 0
AND DP.type_desc NOT IN ('DATABASE_ROLE', 'APPLICATION_ROLE')
AND DP.name NOT IN ('dbo', 'guest', 'INFORMATION_SCHEMA', 'sys',
'MS_DataCollectorInternalUser')
AND NOT (
    DP.name = 'cdc'
    AND (SELECT is_cdc_enabled FROM sys.databases WHERE name = DB_NAME())
= 1
);
```

No rows should be returned to be compliant.

Remediation:

If the orphaned user cannot or should not be matched to an existing or new login using the Microsoft documented process referenced below, run the following T-SQL query in the appropriate database to remove an orphan user:

```
USE <database_name>;  
GO  
DROP USER <username>;
```

References:

1. <https://learn.microsoft.com/en-us/sql/sql-server/failover-clusters/troubleshoot-orphaned-users-sql-server>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.3 <u>Disable Dormant Accounts</u> Delete or disable any dormant accounts after a period of 45 days of inactivity, where supported.			
v7	16.8 <u>Disable Any Unassociated Accounts</u> Disable any account that cannot be associated with a business process or business owner.			

3.4 Ensure SQL Authentication is not used in contained databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Contained databases do not enforce password complexity rules for SQL Authenticated users.

Rationale:

The absence of an enforced password policy may increase the likelihood of a weak credential being established in a contained database.

Impact:

While contained databases provide flexibility in relocating databases to different instances and different environments, this must be balanced with the consideration that no password policy mechanism exists for SQL Authenticated users in contained databases.

Audit:

Execute the following T-SQL in each contained database to find database users that are using SQL authentication:

```
SELECT name AS DBUser
FROM sys.database_principals
WHERE name NOT IN ('dbo','Information_Schema','sys','guest')
AND type IN ('U','S','G')
AND authentication_type = 2;
GO
```

No rows should be returned to be compliant.

Remediation:

Leverage Windows Authenticated users in contained databases.

Default Value:

SQL Authenticated users (**USER WITH PASSWORD** authentication) are allowed in contained databases.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/databases/security-best-practices-with-contained-databases>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.6 <u>Centralize Account Management</u> Centralize account management through a directory or identity service.			
v7	16.2 <u>Configure Centralized Point of Authentication</u> Configure access for all accounts through as few centralized points of authentication as possible, including network, security, and cloud systems.			

3.5 Ensure the SQL Server's MSSQL Service Account is Not an Administrator (Manual)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The service account and/or service SID used by the **MSSQLSERVER** service for a default instance or **MSSQL\$<InstanceName>** service for a named instance should not be a member of the Windows Administrator group either directly or indirectly (via a group). This also means that the account known as **LocalSystem** (aka **NT AUTHORITY\SYSTEM**) should not be used for any of the SQL Server services as this account has higher privileges than any of the SQL Server services require.

Rationale:

Following the principle of least privilege, the service account should have no more privileges than required to do its job. For SQL Server services, the SQL Server Setup will assign the required permissions directly to the service **SID**. No additional permissions or privileges should be necessary.

Impact:

The **SQL Server Configuration Manager** tool should always be used to change the SQL Server's service account. This will ensure that the account has the necessary privileges. If the service needs access to resources other than the standard Microsoft defined directories and registry, then additional permissions may need to be granted separately to those resources.

Audit:

Verify that the service account for running MSSQL is not LocalSystem, by executing the below query:

```
SELECT servicename, service_account
FROM sys.dm_server_services
WHERE servicename LIKE 'SQL Server (%)' AND service_account IN ( 'NT
AUTHORITY\SYSTEM', 'LocalSystem', 'NT AUTHORITY\LocalSystem' );
```

If a record is returned then this is a finding and requires remediation. If no record is returned, verify that the account used is not part of the Local Administrators group.

Remediation:

In the case where **LocalSystem** is used, use **SQL Server Configuration Manager** to change to a less privileged account. Otherwise, remove the account or service **SID** from the Administrators group. You may need to run the **SQL Server Configuration Manager** if underlying permissions had been changed or if **SQL Server Configuration Manager** was not originally used to set the service account.

Default Value:

By default, the Service Account (or Service **SID**) is not a member of the Administrators group.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.4 Restrict Administrator Privileges to Dedicated Administrator Accounts Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	4.3 Ensure the Use of Dedicated Administrative Accounts Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

3.6 Ensure the SQL Server's SQLAgent Service Account is Not an Administrator (Manual)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The service account and/or service **SID** used by the **SQLSERVERAGENT** service for a default instance or **SQLAGENT\$<InstanceName>** service for a named instance should not be a member of the Windows Administrator group either directly or indirectly (via a group). This also means that the account known as **LocalSystem** (AKA **NT AUTHORITY\SYSTEM**) should not be used for the **SQLAGENT** service as this account has higher privileges than the SQL Server service requires.

Rationale:

Following the principle of least privilege, the service account should have no more privileges than required to do its job. For SQL Server services, the SQL Server Setup will assign the required permissions directly to the service **SID**. No additional permissions or privileges should be necessary.

Impact:

The **SQL Server Configuration Manager** tool should always be used to change the SQL Server's service account. This will ensure that the account has the necessary privileges. If the service needs access to resources other than the standard Microsoft-defined directories and registry, then additional permissions may need to be granted separately to those resources.

If using the auto restart feature, then the **SQLAGENT** service must be an Administrator.

Audit:

Verify that the service account (in case of a local or AD account) and service **SID** are not members of the Windows Administrators group.

Verify that the service account for running SQLAgent is not LocalSystem, by executing the below query:

```
SELECT servicename, service_account
FROM sys.dm_server_services
WHERE servicename LIKE 'SQL Server Agent (%)' AND service_account IN ( 'NT AUTHORITY\SYSTEM', 'LocalSystem', 'NT AUTHORITY\LocalSystem' );
```

If a record is returned then this is a finding and requires remediation. If no record is returned, verify that the account used is not part of the Local Administrators group.

Remediation:

In the case where **LocalSystem** is used, use **SQL Server Configuration Manager** to change to a less privileged account. Otherwise, remove the account or service **SID** from the Administrators group. You may need to run the **SQL Server Configuration Manager** if underlying permissions had been changed or if **SQL Server Configuration Manager** was not originally used to set the service account.

Default Value:

By default, the Service Account (or Service **SID**) is not a member of the Administrators group.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.4 Restrict Administrator Privileges to Dedicated Administrator Accounts Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	4.3 Ensure the Use of Dedicated Administrative Accounts Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

3.7 Ensure the SQL Server's Full-Text Service Account is Not an Administrator (Manual)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The service account and/or service **SID** used by the **MSSQLFDLauncher** service for a default instance or **MSSQLFDLauncher\$<InstanceName>** service for a named instance should not be a member of the Windows Administrator group either directly or indirectly (via a group). This also means that the account known as **LocalSystem** (aka **NT AUTHORITY\SYSTEM**) should not be used for the Full-Text service as this account has higher privileges than the SQL Server service requires.

Rationale:

Following the principle of least privilege, the service account should have no more privileges than required to do its job. For SQL Server services, the SQL Server Setup will assign the required permissions directly to the service **SID**. No additional permissions or privileges should be necessary.

Impact:

The **SQL Server Configuration Manager** tool should always be used to change the SQL Server's service account. This will ensure that the account has the necessary privileges. If the service needs access to resources other than the standard Microsoft-defined directories and registry, then additional permissions may need to be granted separately to those resources.

Audit:

Verify that the service account (in case of a local or AD account) and service **SID** are not members of the Windows Administrators group.

Remediation:

In the case where **LocalSystem** is used, use **SQL Server Configuration Manager** to change to a less privileged account. Otherwise, remove the account or service **SID** from the Administrators group. You may need to run the **SQL Server Configuration Manager** if underlying permissions had been changed or if **SQL Server Configuration Manager** was not originally used to set the service account.

Default Value:

By default, the Service Account (or Service **SID**) is not a member of the Administrators group.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.4 <u>Restrict Administrator Privileges to Dedicated Administrator Accounts</u> Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	4.3 <u>Ensure the Use of Dedicated Administrative Accounts</u> Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

3.8 Ensure only the default permissions specified by Microsoft are granted to the public server role (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

public is a special fixed server role containing all logins. Unlike other fixed server roles, permissions can be changed for the **public** role. In keeping with the principle of least privileges, the **public** server role should not be used to grant permissions at the server scope as these would be inherited by all users.

Rationale:

Every SQL Server login belongs to the **public** role and cannot be removed from this role. Therefore, any permissions granted to this role will be available to all logins unless they have been explicitly denied to specific logins or user-defined server roles.

Impact:

When the extraneous permissions are revoked from the **public** server role, access may be lost unless the permissions are granted to the explicit logins or to user-defined server roles containing the logins which require the access.

Audit:

Use the following syntax to determine if extra permissions have been granted to the **public** server role.

```
SELECT *
FROM master.sys.server_permissions
WHERE (grantee_principal_id = SUSER_SID(N'public') and state_desc LIKE
'GRANT%')
AND NOT (state_desc = 'GRANT' and [permission_name] = 'VIEW ANY DATABASE' and
class_desc = 'SERVER')
AND NOT (state_desc = 'GRANT' and [permission_name] = 'CONNECT' and
class_desc = 'ENDPOINT' and major_id = 2)
AND NOT (state_desc = 'GRANT' and [permission_name] = 'CONNECT' and
class_desc = 'ENDPOINT' and major_id = 3)
AND NOT (state_desc = 'GRANT' and [permission_name] = 'CONNECT' and
class_desc = 'ENDPOINT' and major_id = 4)
AND NOT (state_desc = 'GRANT' and [permission_name] = 'CONNECT' and
class_desc = 'ENDPOINT' and major_id = 5);
```

No rows should be returned to be compliant.

Remediation:

1. Add the extraneous permissions found in the Audit query results to the specific logins to user-defined server roles which require the access.
2. Revoke the *<permission_name>* from the *public* role as shown below

```
USE [master]
GO
REVOKE <permission_name> FROM public;
GO
```

Default Value:

By default, the *public* server role is granted **VIEW ANY DATABASE** permission and the **CONNECT** permission on the default endpoints (**TSQL Local Machine**, **TSQL Named Pipes**, **TSQL Default TCP**, **TSQL Default VIA**). The **VIEW ANY DATABASE** permission allows all logins to see database metadata, unless explicitly denied.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/server-level-roles>
2. <https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/server-level-roles#permissions-of-fixed-server-roles>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

3.9 Ensure Windows **BUILTIN** groups are not SQL Logins (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Prior to SQL Server 2008, the **BUILTIN\Administrators** group was added as a SQL Server login with sysadmin privileges during installation by default. Best practices promote creating an Active Directory level group containing approved DBA staff accounts and using this controlled AD group as the login with sysadmin privileges. The AD group should be specified during SQL Server installation and the **BUILTIN\Administrators** group would therefore have no need to be a login.

Rationale:

The **BUILTIN** groups (Administrators, Everyone, Authenticated Users, Guests, etc.) generally contain very broad memberships which would not meet the best practice of ensuring only the necessary users have been granted access to a SQL Server instance. These groups should not be used for any level of access into a SQL Server Database Engine instance.

Impact:

Before dropping the **BUILTIN** group logins, ensure that alternative AD Groups or Windows logins have been added with equivalent permissions. Otherwise, the SQL Server instance may become totally inaccessible.

Audit:

Use the following syntax to determine if any **BUILTIN** groups or accounts have been added as SQL Server Logins.

```
SELECT pr.[name], pe.[permission_name], pe.[state_desc]
FROM sys.server_principals pr
JOIN sys.server_permissions pe
ON pr.principal_id = pe.grantee_principal_id
WHERE pr.name like 'BUILTIN%';
```

No rows should be returned to be compliant. On an AWS RDS instance if only **[BUILTIN]\Administrators** is returned, this is considered a pass and therefore compliant.

Remediation:

1. For each **BUILTIN** login, if needed create a more restrictive AD group containing only the required user accounts.
2. Add the AD group or individual Windows accounts as a SQL Server login and grant it the permissions required.
3. Drop the **BUILTIN** login using the syntax below after replacing *<name>* in **[BUILTIN\<name>]**.

```
USE [master]
GO
DROP LOGIN [BUILTIN\<name>]
GO
```

Default Value:

By default, no **BUILTIN** groups are added as SQL logins.

Additional Information:

In AWS RDS instances **[BUILTIN]\Administrators** can't be dropped. Dropping **[Builtin]\Administrators** is blocked in AWS RDS by the server-level trigger **rds_drop_login_trigger**.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

3.10 Ensure Windows local groups are not SQL Logins (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Local Windows groups should not be used as logins for SQL Server instances.

Rationale:

Allowing local Windows groups as SQL Logins provides a loophole whereby anyone with OS level administrator rights (and no SQL Server rights) could add users to the local Windows groups and thereby give themselves or others access to the SQL Server instance.

Impact:

Before dropping the local group logins, ensure that alternative AD Groups or Windows logins have been added with equivalent permissions. Otherwise, the SQL Server instance may become totally inaccessible.

Audit:

Use the following syntax to determine if any local groups have been added as SQL Server Logins.

```
SELECT pr.[name] AS LocalGroupName, pe.[permission_name], pe.[state_desc]
FROM sys.server_principals pr
JOIN sys.server_permissions pe
ON pr.[principal_id] = pe.[grantee_principal_id]
WHERE pr.[type_desc] = 'WINDOWS_GROUP'
AND pr.[name] like CAST(SERVERPROPERTY('MachineName') AS nvarchar) + '%';
```

No rows should be returned to be compliant.

Remediation:

1. For each **LocalGroupName** login, if needed create an equivalent AD group containing only the required user accounts.
2. Add the AD group or individual Windows accounts as a SQL Server login and grant it the permissions required.
3. Drop the **LocalGroupName** login using the syntax below after replacing **<name>**.

```
USE [master]
GO
DROP LOGIN [<name>]
GO
```

Default Value:

By default, no local groups are added as SQL logins.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

3.11 Ensure the public role in the msdb database is not granted access to SQL Agent proxies (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **public** database role contains every user in the **msdb** database. SQL Agent proxies define a security context in which a job step can run.

Rationale:

Granting access to SQL Agent proxies for the **public** role would allow all users to utilize the proxy which may have high privileges. This would likely break the principle of least privileges.

Impact:

Before revoking the **public** role from the proxy, ensure that alternative logins or appropriate user-defined database roles have been added with equivalent permissions. Otherwise, SQL Agent job steps dependent upon this access will fail.

Audit:

Use the following syntax to determine if access to any proxies have been granted to the **msdb** database's **public** role.

```
USE [msdb]
GO
SELECT sp.name AS proxyname
FROM dbo.sysproxylogin spl
JOIN sys.database_principals dp
ON dp.sid = spl.sid
JOIN sysproxies sp
ON sp.proxy_id = spl.proxy_id
WHERE principal_id = USER_ID('public');
GO
```

No rows should be returned to be compliant.

Remediation:

1. Ensure the required security principals are explicitly granted access to the proxy (use `sp_grant_login_to_proxy`).
2. Revoke access to the `<proxyname>` from the `public` role.

```
USE [msdb]
GO
EXEC dbo.sp_revoke_login_from_proxy @name = N'public', @proxy_name =
N'<proxyname>';
GO
```

Default Value:

By default, the `msdb public` database role does not have access to any proxy.

References:

1. <https://learn.microsoft.com/en-US/sql/ssms/agent/create-a-sql-server-agent-proxy>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 Configure Data Access Control Lists Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 Protect Information through Access Control Lists Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

3.12 Ensure the 'SYSADMIN' Role is Limited to Administrative or Built-in Accounts (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

The **SYSADMIN** role is the highest privileged server-level role in SQL Server database engine. Moreover, by design built-in accounts by default are granted permission to this server-level role by Microsoft design so database engine works as expected. The following virtual accounts / Service SIDs are default members of SYSADMIN: NT SERVICE\SQLWriter NT SERVICE\Winmgmt NT SERVICE\MSSQLSERVER (Used by the SQL database engine service) NT SERVICE\SQLSERVERAGENT(Used by the SQL Agent service) This means that the service accounts for the SQL Database Engine and SQL Agent does not need to, and should not have , their specific service accounts added to the SYSADMIN group separately, as it is not needed.

The built-in database **sa** account and service accounts are automatically created during SQL Server installation are required to be granted **SYSADMIN** role. DBA's can create accounts with **SYSADMIN** role for support and administration. Such accounts should be limited as well as protected using strict access and authorization restrictions.

Rationale:

This will greatly reduces the attack surface, as only limited and specific accounts will be granted **SYSADMIN** role. So, attackers can't break into the database system with highly privileged accounts.

Audit:

Execute this SQL query to find current service accounts running your SQL Server Engine with **SYSADMIN** role permission:

```
SELECT    distinct(name),type_desc
FROM      master.sys.server_principals a , sys.dm_server_services b
WHERE     IS_SRVROLEMEMBER ('sysadmin',name) = 1 and a.name=b.service_account;
```

Execute this SQL query to list all SQL Server instance principles with SYSADMIN role granted to them:

```

SELECT  distinct(name),type_desc
FROM    master.sys.server_principals
WHERE   IS_SRVROLEMEMBER ('sysadmin',name) = 1
AND name not in (
'NT SERVICE\SQLWriter',
'NT SERVICE\Winmgmt',
'NT SERVICE\MSSQLSERVER',
'NT SERVICE\SQLSERVERAGENT'
);

```

If any un-allowed accounts have **SYSADMIN** role, this is a fail.

Remediation:

Remove any un-allowed SQL Server accounts which are granted **SYSADMIN** role using this query:

```
ALTER SERVER ROLE SYSADMIN DROP MEMBER <account>;
```

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.4 <u>Restrict Administrator Privileges to Dedicated Administrator Accounts</u> Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	4.3 <u>Ensure the Use of Dedicated Administrative Accounts</u> Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

3.13 Ensure no admin role membership in MSDB database (Automated)

Profile Applicability:

- Level 1 - Database Engine

Description:

Based on Microsoft design an account with **DB_OWNER** can elevate permissions to **SYSADMIN**

Rationale:

MSDB must be configured with the **TRUSTWORTHY** flag **ON** to work properly. If the **TRUSTWORTHY** setting is set to **ON**, and if the owner of the database is a member of a group that has administrative credentials, such as the sysadmin group (for example the default **sa** login), the database owner can then be able to create and run unsafe assemblies that can compromise the instance of the SQL Server, as well as run code to elevate his privileges to **SYSADMIN**

Audit:

```
USE [msdb]

SELECT * FROM sys.database_role_members AS drm
INNER JOIN sys.database_principals AS r
    ON drm.role_principal_id = r.principal_id
INNER JOIN sys.database_principals AS m
    ON drm.member_principal_id = m.principal_id
WHERE r.name in ('db_owner', 'db_securityadmin', 'db_ddladmin',
'db_datawriter') and m.name <>'dbo';

GO
```

No rows should be returned to be compliant.

Remediation:

```
USE [msdb]
GO

ALTER ROLE [db_owner] DROP MEMBER <username>;
```

Default Value:

Default value is that only dbo user is member of **db_owner** role in **MSDB** database

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/trustworthy-database-property?view=sql-server-ver16>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>5.4 Restrict Administrator Privileges to Dedicated Administrator Accounts</u> Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	<u>4.3 Ensure the Use of Dedicated Administrative Accounts</u> Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

3.14 Ensure 'Control Server' permission is not granted (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

The Control Server permission is a highly privileged server-level permission in SQL Server database engine. Moreover, even though this permission is not as powerful as **SYSADMIN** role, it has almost most of the permissions of **SYSADMIN** role. Also, database logins granted control server permission can easily elevate to **SYSADMIN** role through different exploit techniques which is a serious security problem.

Rationale:

This will greatly reduce attack surface, as accounts that get compromised by attackers with Control Server permission can escalate to **SYSADMIN** role. So, attackers can't break into the database system with highly privileged accounts. Revoking control server permission will prevent privilege elevation attacks.

Audit:

Execute this SQL query to list all SQL Server logins with Control Server permission:

```

SELECT
    sys.server_permissions.type,
    sys.server_permissions.permission_name,
    sys.server_permissions.state,
    sys.server_permissions.state_desc,
    granteeserverprincipal.name AS grantee_principal_name,
    granteeserverprincipal.type_desc AS grantee_principal_type_desc,
    grantorserverprincipal.name AS grantor_name,
    CASE
        WHEN sys.server_permissions.state = N'W'
            THEN N'GRANT'
        ELSE sys.server_permissions.state_desc
    END + N' ' + sys.server_permissions.permission_name COLLATE
SQL_Latin1_General_CP1_CI_AS + N' TO ' +
QUOTENAME(granteeserverprincipal.name) AS permissionstatement
FROM sys.server_principals AS granteeserverprincipal
INNER JOIN sys.server_permissions
    ON sys.server_permissions.grantee_principal_id =
granteeserverprincipal.principal_id
INNER JOIN sys.server_principals AS grantorserverprincipal
    ON grantorserverprincipal.principal_id =
sys.server_permissions.grantor_principal_id
WHERE sys.server_permissions.permission_name='CONTROL SERVER'
AND granteeserverprincipal.name <> '##MS_PolicySigningCertificate##';

```

Remediation:

Remove any un-allowed SQL Server accounts which are granted Control Server using this query:

```

use [master]
GO
REVOKE CONTROL SERVER FROM <login_name>

```

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/permissions-database-engine>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>5.4 Restrict Administrator Privileges to Dedicated Administrator Accounts</u> Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	<u>4.3 Ensure the Use of Dedicated Administrative Accounts</u> Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

4 Password Policies

This section contains recommendations related to SQL Server's password policies.

4.1 Ensure 'MUST_CHANGE' Option is set to 'ON' for All SQL Authenticated Logins (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

Whenever this option is set to **ON**, SQL Server will prompt for an updated password the first time the new or altered login is used.

Rationale:

Enforcing a password change after a reset or new login creation will prevent the account administrators or anyone accessing the initial password from misuse of the SQL login created without being noticed.

Impact:

CHECK_EXPIRATION and **CHECK_POLICY** options must both be **ON**. End users must have the means (application) to change the password when forced.

Audit:

Using SQL Server Management Studio

1. Open **SQL Server Management Studio**.
2. Open **Object Explorer** and connect to the target instance.
3. Navigate to the **Logins** tab in **Object Explorer** and expand. Right click on the desired login and select **Properties**.
4. Verify the User must change password at next login checkbox is checked.

Using T-SQL

```
SELECT name ,CAST(LOGINPROPERTY(log.name, N'IsMustChange') AS bit) AS  
[MustChangePassword],*  
FROM sys.server_principals AS log  
WHERE type = 'S' and CAST(LOGINPROPERTY(log.name, N'IsMustChange') AS bit) =  
1
```

Returns "1" row for each new Login that has not been logged in to yet, and is correctly configured

Note: This audit procedure is only applicable immediately after the login has been created or altered to force the password change. Once the password is changed, there is no way to know specifically that this option was the forcing mechanism behind a password change.

Remediation:

Set the **MUST_CHANGE** option for SQL Authenticated logins when creating a login initially:

```
CREATE LOGIN <login_name> WITH PASSWORD = '<password_value>' MUST_CHANGE,  
CHECK_EXPIRATION = ON, CHECK_POLICY = ON;
```

Set the **MUST_CHANGE** option for SQL Authenticated logins when resetting a password:

```
ALTER LOGIN <login_name> WITH PASSWORD = '<new_password_value>' MUST_CHANGE;
```

Default Value:

ON when creating a new login via the SSMS GUI. **OFF** when creating a new login using T-SQL **CREATE LOGIN** unless the **MUST_CHANGE** option is explicitly included along with **CHECK_EXPIRATION = ON**.

References:

1. <https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-login-transact-sql>
2. <https://docs.microsoft.com/en-us/sql/t-sql/statements/create-login-transact-sql>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.7 Manage Default Accounts on Enterprise Assets and Software Manage default accounts on enterprise assets and software, such as root, administrator, and other pre-configured vendor accounts. Example implementations can include: disabling default accounts or making them unusable.			
v7	4.2 Change Default Passwords Before deploying any new asset, change all default passwords to have values consistent with administrative level accounts.			

4.2 Ensure 'CHECK_EXPIRATION' Option is set to 'ON' for All SQL Authenticated Logins Within the Sysadmin Role (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Applies the same password expiration policy used in Windows to passwords used inside SQL Server.

Rationale:

Ensuring SQL logins comply with the secure password policy applied by the Windows Server Benchmark will ensure the passwords for SQL logins with **sysadmin** privileges are changed on a frequent basis to help prevent compromise via a brute force attack. **CONTROL SERVER** is an equivalent permission to **sysadmin** and logins with that permission should also be required to have expiring passwords.

Impact:

This is a mitigating recommendation for systems which cannot follow the recommendation to use only Windows Authenticated logins.

Regarding limiting this rule to only logins with **sysadmin** and **CONTROL SERVER** privileges, there are too many cases of applications that run with less than sysadmin level privileges that have hard-coded passwords or effectively hard-coded passwords (whatever is set the first time is nearly impossible to change). There are several line-of-business applications that are considered best of breed which have this failing.

Also, keep in mind that the password policy is taken from the computer's local policy, which is taken from the Default Domain Policy setting. Many organizations have a different password policy regarding the service accounts. These are handled in AD by setting the account's password to not expire and having some other process track when the password needs to be changed. With this second control in place, this is perfectly acceptable from an audit perspective. If you treat a SQL Server login as a service account, then you have to do the same. This ensures that the password change happens during a communicated downtime window and not arbitrarily.

Audit:

Run the following T-SQL statement to find **sysadmin** or equivalent logins with **CHECK_EXPIRATION = OFF**.

```
SELECT l.[name], 'sysadmin membership' AS 'Access_Method'
FROM sys.sql_logins AS l
WHERE IS_SRVROLEMEMBER('sysadmin',name) = 1
AND l.is_expiration_checked <> 1 AND l.is_disabled = 0
UNION ALL
SELECT l.[name], 'CONTROL SERVER' AS 'Access_Method'
FROM sys.sql_logins AS l
JOIN sys.server_permissions AS p
ON l.principal_id = p.grantee_principal_id
WHERE p.type = 'CL' AND p.state IN ('G', 'W')
AND l.is_expiration_checked <> 1 AND l.is_disabled = 0;
```

No rows should be returned to be compliant. On AWS RDS instances only returning the account **rdsa** is considered a pass and therefore compliant.

Remediation:

For each **<login_name>** found by the Audit Procedure, execute the following T-SQL statement:

```
ALTER LOGIN [<login_name>] WITH CHECK_EXPIRATION = ON;
```

Default Value:

CHECK_EXPIRATION is **ON** by default when using SSMS to create a SQL authenticated login.

CHECK_EXPIRATION is **OFF** by default when using T-SQL **CREATE LOGIN** syntax without specifying the **CHECK_EXPIRATION** option.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/password-policy>

Additional Information:

The **rdsa** account created by AWS RDS cannot be altered. It has the **Sysadmin** role but its password cannot be changed to allow it to expire.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.3 <u>Disable Dormant Accounts</u> Delete or disable any dormant accounts after a period of 45 days of inactivity, where supported.			
v7	16.10 <u>Ensure All Accounts Have An Expiration Date</u> Ensure that all accounts have an expiration date that is monitored and enforced.			

4.3 Ensure 'CHECK_POLICY' Option is set to 'ON' for All SQL Authenticated Logins (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Applies the same password complexity policy used in Windows to passwords used inside SQL Server.

Rationale:

Ensure SQL authenticated login passwords comply with the secure password policy applied by the Windows Server Benchmark so that they cannot be easily compromised via brute force attack.

Impact:

This is a mitigating recommendation for systems which cannot follow the recommendation to use only Windows Authenticated logins.

Weak passwords can lead to compromised systems. SQL Server authenticated logins will utilize the password policy set in the computer's local policy, which is typically set by the Default Domain Policy setting.

The setting is only enforced when the password is changed. This setting does not force existing weak passwords to be changed.

Audit:

Use the following code snippet to determine the status of SQL Logins and if their password complexity is enforced.

```
SELECT name, is_disabled
FROM sys.sql_logins
WHERE is_policy_checked = 0;
```

The **is_policy_checked** value of **0** indicates that the **CHECK_POLICY** option is **OFF**; value of **1** is **ON**. If **is_disabled** value is **1**, then the login is disabled and unusable. If no rows are returned then either no SQL Authenticated logins exist or they all have **CHECK_POLICY ON**.

Remediation:

For each *<login_name>* found by the Audit Procedure, execute the following T-SQL statement:

```
ALTER LOGIN [<login_name>] WITH CHECK_POLICY = ON;
```

Note: In the case of AWS RDS do not perform this remediation for the Master account.

Default Value:

CHECK_POLICY is ON

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/password-policy>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	5.2 Use Unique Passwords Use unique passwords for all enterprise assets. Best practice implementation includes, at a minimum, an 8-character password for accounts using MFA and a 14-character password for accounts not using MFA.			
v7	4.4 Use Unique Passwords Where multi-factor authentication is not supported (such as local administrator, root, or service accounts), accounts will use passwords that are unique to that system.			

5 Auditing and Logging

This section contains recommendations related to SQL Server's audit and logging mechanisms.

5.1 Ensure 'Maximum number of error log files' is set to greater than or equal to '12' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

SQL Server error log files must be protected from loss. The log files must be backed up before they are overwritten. Retaining more error logs helps prevent loss from frequent recycling before backups can occur.

Rationale:

The SQL Server error log contains important information about major server events and login attempt information as well.

Impact:

Once the max number of error logs is reached, the oldest error log file is deleted each time SQL Server restarts or `sp_cycle_errorlog` is executed.

Audit:

Perform either the GUI or T-SQL method shown:

GUI Method

1. Open **SQL Server Management Studio**.
2. Open **Object Explorer** and connect to the target instance.
3. Navigate to the **Management** tab in **Object Explorer** and expand. Right click on the **SQL Server Logs** file and select **Configure**.
4. Verify the **Limit the number of error log files before they are recycled** checkbox is checked
5. Verify the **Maximum number of error log files** is greater than or equal to **12**, if a limit is configured.

T-SQL Method

Run the following T-SQL. The `NumberOfLogFiles` returned should be greater than or equal to **12**, or equal to **-1** if no limit is configured.

```
DECLARE @NumErrorLogs int;
EXEC master.sys.xp_instance_regread
N'HKEY_LOCAL_MACHINE',
N'Software\Microsoft\MSSQLServer\MSSQLServer',
N'NumErrorLogs',
@NumErrorLogs OUTPUT;
SELECT ISNULL(@NumErrorLogs, -1) AS [NumberOfLogFiles];
```

Remediation:

Adjust the number of logs to prevent data loss. The default value of **6** may be insufficient for a production environment. Perform either the GUI or T-SQL method shown:

GUI Method

1. Open **SQL Server Management Studio**.
2. Open **Object Explorer** and connect to the target instance.
3. Navigate to the **Management** tab in **Object Explorer** and expand. Right click on the **SQL Server Logs** file and select **Configure**
4. Check the **Limit the number of error log files before they are recycled**
5. Set the **Maximum number of error log files** to greater than or equal to **12**

T-SQL Method

Run the following T-SQL to change the number of error log files, replace **<NumberGreaterThanOrEqualTo11>** with your desired number of error log files:

```
EXEC master.sys.xp_instance_regwrite
N'HKEY_LOCAL_MACHINE',
N'Software\Microsoft\MSSQLServer\MSSQLServer',
N'NumErrorLogs',
REG_DWORD,
<NumberGreaterThanOrEqualTo11>;
```

Default Value:

6 SQL Server error log files in addition to the current error log file are retained by default.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/scm-services-configure-sql-server-error-logs>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.3 <u>Ensure Adequate Audit Log Storage</u> Ensure that logging destinations maintain adequate storage to comply with the enterprise's audit log management process.			
v7	6.4 <u>Ensure adequate storage for logs</u> Ensure that all systems that store logs have adequate storage space for the logs generated.			

5.2 Ensure 'Default Trace Enabled' Server Configuration Option is set to '1' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

The default trace provides audit logging of database activity including account creations, privilege elevation and execution of DBCC commands.

Rationale:

Default trace provides valuable audit information regarding security-related activities on the server.

Audit:

Run the following T-SQL command:

```
SELECT name,  
       CAST(value as int) as value_configured,  
       CAST(value_in_use as int) as value_in_use  
FROM sys.configurations  
WHERE name = 'default trace enabled';
```

Both value columns must show **1** to be compliant.

Remediation:

Run the following T-SQL command:

```
EXECUTE sp_configure 'show advanced options', 1;  
RECONFIGURE;  
EXECUTE sp_configure 'default trace enabled', 1;  
RECONFIGURE;  
GO  
EXECUTE sp_configure 'show advanced options', 0;  
RECONFIGURE;
```

Default Value:

1 (on)

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/default-trace-enabled-server-configuration-option>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.2 <u>Collect Audit Logs</u> Collect audit logs. Ensure that logging, per the enterprise's audit log management process, has been enabled across enterprise assets.			
v8	8.5 <u>Collect Detailed Audit Logs</u> Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.			
v7	6.2 <u>Activate audit logging</u> Ensure that local logging has been enabled on all systems and networking devices.			
v7	6.3 <u>Enable Detailed Logging</u> Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.			

5.3 Ensure 'Login Auditing' is set to 'failed logins' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

This setting will record failed authentication attempts for SQL Server logins to the **SQL Server Errorlog**. This is the default setting for SQL Server.

Historically, this setting has been available in all versions and editions of SQL Server. Prior to the availability of **SQL Server Audit**, this was the only provided mechanism for capturing logins (successful or failed).

Rationale:

Capturing failed logins provides key information that can be used to detect\confirm password guessing attacks. Capturing successful login attempts can be used to confirm server access during forensic investigations, but using this audit level setting to also capture successful logins creates excessive noise in the **SQL Server Errorlog** which can hamper a DBA trying to troubleshoot problems. Elsewhere in this benchmark, we recommend using the newer lightweight SQL Server Audit feature to capture both successful and failed logins.

Impact:

At a minimum, we want to ensure failed logins are captured in order to detect if an adversary is attempting to brute force passwords or otherwise attempting to access a SQL Server improperly.

Changing the setting requires a restart of the SQL Server service.

Audit:

```
EXEC xp_loginconfig 'audit level';
```

A **config_value** of **failure** indicates a server login auditing setting of **Failed logins only**. If a **config_value** of **all** appears, then both failed and successful logins are being logged. Both settings should also be considered valid, but as mentioned capturing successful logins using this method creates lots of noise in the **SQL Server Errorlog**.

Remediation:

Perform either the GUI or T-SQL method shown:

GUI Method

1. Open **SQL Server Management Studio**.

2. Right click the target instance and select **Properties** and navigate to the **Security** tab.
3. Select the option **Failed logins only** under the **Login Auditing** section and click **OK**.
4. Restart the SQL Server instance.

T-SQL Method

1. Run:

```
EXEC xp_instance_regwrite N'HKEY_LOCAL_MACHINE',
N'Software\Microsoft\MSSQLServer\MSSQLServer', N'AuditLevel',
REG_DWORD, 2
```

2. Restart the SQL Server instance.

Default Value:

By default, only failed login attempts are captured.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/server-properties-security-page>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.1 <u>Establish and Maintain an Audit Log Management Process</u> Establish and maintain an audit log management process that defines the enterprise's logging requirements. At a minimum, address the collection, review, and retention of audit logs for enterprise assets. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	16.13 <u>Alert on Account Login Behavior Deviation</u> Alert when users deviate from normal login behavior, such as time-of-day, workstation location and duration.			

5.4 Ensure 'SQL Server Audit' is set to capture both 'failed' and 'successful logins' (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

SQL Server Audit is capable of capturing both failed and successful logins and writing them to one of three places: the application event log, the security event log, or the file system. We will use it to capture any login attempt to SQL Server, as well as any attempts to change audit policy, changes in privileged role memberships and changes to server settings. This will also serve to be a second source to record failed login attempts.

Rationale:

By utilizing Audit instead of the traditional setting under the Security tab to capture successful logins, we reduce the noise in the **ERRORLOG**. This keeps it smaller and easier to read for DBAs who are attempting to troubleshoot issues with the SQL Server. Also, the Audit object can write to the security event log, though this requires operating system configuration. This gives an additional option for where to store login events, especially in conjunction with an SIEM.

Impact:

With the previous recommendation, only failed logins are captured. If the Audit object is not implemented with the appropriate setting, SQL Server will not capture successful logins, which might prove of use for forensics.

Audit:

For AWS RDS Instances, if RDS has not been configured to write to an S3 bucket, this is a fail.

Run the following T-SQL command:

```

SELECT
  S.name AS 'Audit Name'
  , CASE S.is_state_enabled
  WHEN 1 THEN 'Y'
  WHEN 0 THEN 'N' END AS 'Audit Enabled'
  , S.type_desc AS 'Write Location'
  , SA.name AS 'Audit Specification Name'
  , CASE SA.is_state_enabled
  WHEN 1 THEN 'Y'
  WHEN 0 THEN 'N' END AS 'Audit Specification Enabled'
  , SAD.audit_action_name
  , SAD.audited_result
FROM sys.server_audit_specification_details AS SAD
JOIN sys.server_audit_specifications AS SA
ON SAD.server_specification_id = SA.server_specification_id
JOIN sys.server_audits AS S
ON SA.audit_guid = S.audit_guid
WHERE SAD.audit_action_id IN ('CNAU', 'LGFL', 'LGSD', 'ADDP', 'ADSP', 'OPSV')
or (SAD.audit_action_id IN ('DAGS', 'DAGF') and (select count(*) from
sys.databases where containment=1) > 0);

```

The result set should contain the following rows, one for each of the following **audit_action_names**:

- **AUDIT_CHANGE_GROUP**
- **FAILED_LOGIN_GROUP**
- **SUCCESSFUL_LOGIN_GROUP**
- **DATABASE_ROLE_MEMBER_CHANGE_GROUP**
- **SERVER_ROLE_MEMBER_CHANGE_GROUP**
- **SERVER_OPERATION_GROUP**

The result set should also contain these 2 rows if there are contained databases

- **SUCCESSFUL_DATABASE_AUTHENTICATION_GROUP**
- **FAILED_DATABASE_AUTHENTICATION_GROUP**

Both the Audit and Audit specification should be enabled and the **audited_result** should include both success and failure.

Remediation:

For AWS RDS Instances, please refer to the documentation for configuring SQL Server Audit here: [SQL Server Audit](#)

Perform either the GUI or T-SQL method shown:

GUI Method

1. Expand the **SQL Server** in **Object Explorer**.
2. Expand the **Security Folder**
3. Right-click on the **Audits** folder and choose **New Audit...**

4. Specify a name for the **Server Audit**.
5. Specify the audit destination details and then click **OK** to save the **Server Audit**.
6. Right-click on **Server Audit Specifications** and choose **New Server Audit Specification...**
7. Name the **Server Audit Specification**
8. Select the just created **Server Audit** in the **Audit** drop-down selection.
9. Click the drop-down under **Audit Action Type** and select **AUDIT_CHANGE_GROUP**.
10. Click the new drop-down **Audit Action Type** and select **FAILED_LOGIN_GROUP**.
11. Click the new drop-down under **Audit Action Type** and select **SUCCESSFUL_LOGIN_GROUP**.
12. Click the new drop-down under **Audit Action Type** and select **DATABASE_ROLE_MEMBER_CHANGE_GROUP**.
13. Click the new drop-down under **Audit Action Type** and select **SERVER_ROLE_MEMBER_CHANGE_GROUP**.
14. Click the new drop-down under **Audit Action Type** and select **SERVER_OPERATION_GROUP**.
15. Click the new drop-down under **Audit Action Type** and select **SUCCESSFUL_DATABASE_AUTHENTICATION_GROUP**.
16. Click the new drop-down under **Audit Action Type** and select **FAILED_DATABASE_AUTHENTICATION_GROUP**.
17. Click **OK** to save the **Server Audit Specification**.
18. Right-click on the new **Server Audit Specification** and select **Enable Server Audit Specification**.
19. Right-click on the new **Server Audit** and select **Enable Server Audit**.

T-SQL Method

Execute code similar to:

```

USE [master]
GO
CREATE SERVER AUDIT TrackLogins
TO APPLICATION_LOG;
GO
CREATE SERVER AUDIT SPECIFICATION TrackAllLogins
FOR SERVER AUDIT TrackLogins
    ADD (FAILED_LOGIN_GROUP),
    ADD (SUCCESSFUL_LOGIN_GROUP),
    ADD (AUDIT_CHANGE_GROUP),
    ADD (DATABASE_ROLE_MEMBER_CHANGE_GROUP),
    ADD (SERVER_ROLE_MEMBER_CHANGE_GROUP),
    ADD (SERVER_OPERATION_GROUP),
    ADD (SUCCESSFUL_DATABASE_AUTHENTICATION_GROUP),
    ADD (FAILED_DATABASE_AUTHENTICATION_GROUP)
WITH (STATE = ON);
GO
ALTER SERVER AUDIT TrackLogins
WITH (STATE = ON);
GO

```

Note: If the write destination for the Audit object is to be the security event log, see the Books Online topic [Write SQL Server Audit Events to the Security Log](#) and follow the appropriate steps.

Default Value:

By default, there are no audit object tracking login events.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/auditing/create-a-server-audit-and-server-audit-specification>

Additional Information:

If you want to filter out "VIEW SERVER STATE" events from the audit (because it can create extra rows in the log, and you may or may not be interested in that specific event), create your server audit with a filter, to exclude that specific event:

```

CREATE SERVER AUDIT TrackLogins
TO APPLICATION_LOG
WHERE ([audit_id]<>(1414746966));

```

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.5 <u>Collect Detailed Audit Logs</u> Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.			
v7	4.9 <u>Log and Alert on Unsuccessful Administrative Account Login</u> Configure systems to issue a log entry and alert on unsuccessful logins to an administrative account.			

6 Application Development

This section contains recommendations related to developing applications that interface with SQL Server.

6.1 Ensure Database and Application User Input is Sanitized (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

Always validate user input received from a database client or application by testing type, length, format, and range prior to transmitting it to the database server.

Rationale:

Sanitizing user input drastically minimizes risk of SQL injection.

Impact:

Sanitizing user input may require changes to application code or database object syntax. These changes can require applications or databases to be taken temporarily off-line. Any change to TSQL or application code should be thoroughly tested in testing environment before production implementation.

Audit:

Check with the application teams to ensure any database interaction is through the use of stored procedures and not dynamic SQL. Revoke any **INSERT**, **UPDATE**, or **DELETE** privileges to users so that modifications to data must be done through stored procedures. Verify that there's no SQL query in the application code produced by string concatenation.

Remediation:

The following steps can be taken to remediate SQL injection vulnerabilities:

- Review TSQL and application code for SQL Injection
- Only permit minimally privileged accounts to send user input to the server
- Minimize the risk of SQL injection attack by using parameterized commands and stored procedures
- Reject user input containing binary data, escape sequences, and comment characters
- Always validate user input and do not use it directly to build SQL statements

References:

1. https://owasp.org/www-community/attacks/SQL_Injection
2. <https://learn.microsoft.com/en-us/sql/relational-databases/security/sql-injection>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>16.1 Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	<u>18.2 Ensure Explicit Error Checking is Performed for All In-house Developed Software</u> For in-house developed software, ensure that explicit error checking is performed and documented for all input, including for size, data type, and acceptable ranges or formats.			

6.2 Ensure 'CLR Assembly Permission Set' is set to 'SAFE_ACCESS' for All CLR Assemblies (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Setting CLR Assembly Permission Sets to **SAFE_ACCESS** will hinder assemblies from accessing external system resources such as files, the network, environment variables, or the registry.

Rationale:

Assemblies with **EXTERNAL_ACCESS** or **UNSAFE** permission sets can be used to access sensitive areas of the operating system, steal and/or transmit data and alter the state and other protection measures of the underlying Windows Operating System.

Assemblies which are part of core SQL Server Engine (**is_user_defined = 0**) are excluded from this check as they are required for overall system functionality.

Impact:

The remediation measure should first be tested within a test environment prior to production to ensure the assembly still functions as designed with **SAFE** permission setting.

Audit:

Execute the following SQL statement:

```
USE <database_name>;
GO
SELECT name,
       permission_set_desc
FROM sys.assemblies
WHERE is_user_defined = 1
      AND permission_set > 1;
```

No rows should be returned to be compliant.

Remediation:

```
USE <database_name>;  
GO  
ALTER ASSEMBLY <assembly_name> WITH PERMISSION_SET = SAFE;
```

Default Value:

SAFE permission is set by default.

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/clr-integration/security/clr-integration-code-access-security>
2. <https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-assemblies-transact-sql>
3. <https://learn.microsoft.com/en-us/sql/t-sql/statements/alter-assembly-transact-sql>
4. <https://learn.microsoft.com/en-us/sql/relational-databases/clr-integration/security/clr-integration-code-access-security#recommended-permission-settings>

Additional Information:

Per Microsoft documentation, "SQL Server contains CLR assemblies that the database engine uses to provide certain functionality. The **Microsoft.SqlServer.Types** assembly that is included with SQL Server installation appears in the metadata as an **UNSAFE** assembly. This is by design. These assemblies are considered trusted & secure by default."

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 Establish and Maintain a Secure Configuration Process Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	5.1 Establish Secure Configurations Maintain documented, standard security configuration standards for all authorized operating systems and software.			

7 Encryption

These recommendations pertain to encryption-related aspects of SQL Server.

7.1 Ensure 'Symmetric Key encryption algorithm' is set to 'AES_128' or higher in non-system databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Per the Microsoft Best Practices, only the SQL Server AES algorithm options, **AES_128**, **AES_192**, and **AES_256**, should be used for a symmetric key encryption algorithm.

Rationale:

The following algorithms (as referred to by SQL Server) are considered weak or deprecated and should no longer be used in SQL Server: **DES**, **DESX**, **RC2**, **RC4**, **RC4_128**.

Many organizations may accept the Triple DES algorithms (**TDEA**) which use keying options 1 (3 key aka **3TDEA**) or keying option 2 (2 key aka **2TDEA**). In SQL Server, these are referred to as **TRIPLE_DES_3KEY** and **TRIPLE_DES** respectively. Additionally, the SQL Server algorithm named DESX is actually the same implementation as the **TRIPLE_DES_3KEY** option. However, using the DESX identifier as the algorithm type has been deprecated and its usage is now discouraged.

Impact:

Eliminates use of weak and deprecated algorithms which may put a system at higher risk of an attacker breaking the key.

Encrypted data cannot be compressed, but compressed data can be encrypted. If you use compression, you should compress data before encrypting it.

Audit:

Run the following code for each individual user database:

```
USE <database_name>
GO
SELECT db_name() AS Database_Name, name AS Key_Name
FROM sys.symmetric_keys
WHERE algorithm_desc NOT IN ('AES_128', 'AES_192', 'AES_256')
AND db_id() > 4;
GO
```

No rows should be returned to be compliant.

Remediation:

Refer to Microsoft SQL Server Books Online ALTER SYMMETRIC KEY entry:
<https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-symmetric-key-transact-sql>

Default Value:

none

References:

1. <https://learn.microsoft.com/en-us/sql/t-sql/statements/alter-symmetric-key-transact-sql>
2. <https://learn.microsoft.com/en-US/sql/relational-databases/security/encryption/choose-an-encryption-algorithm>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

7.2 Ensure Asymmetric Key Size is set to 'greater than or equal to 2048' in non-system databases (Automated)

Profile Applicability:

- Level 1 - Database Engine
- Level 1 - AWS RDS

Description:

Microsoft Best Practices recommend to use at least a 2048-bit encryption algorithm for asymmetric keys.

Rationale:

The **RSA_2048** encryption algorithm for asymmetric keys in SQL Server is the highest bit-level provided and therefore the most secure available choice (other choices are **RSA_512** and **RSA_1024**).

Impact:

The higher-bit level may result in slower performance, but reduces the likelihood of an attacker breaking the key.

Encrypted data cannot be compressed, but compressed data can be encrypted. If you use compression, you should compress data before encrypting it.

Audit:

Run the following code for each individual user database:

```
USE <database_name>
GO
SELECT db_name() AS Database_Name, name AS Key_Name
FROM sys.asymmetric_keys
WHERE key_length < 2048
AND db_id() > 4;
GO
```

No rows should be returned to be compliant.

Remediation:

Refer to Microsoft SQL Server Books Online ALTER ASYMMETRIC KEY entry:
<https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-asymmetric-key-transact-sql>

Default Value:

None

References:

1. <https://learn.microsoft.com/en-us/sql/t-sql/statements/alter-asymmetric-key-transact-sql>
2. <https://learn.microsoft.com/en-US/sql/relational-databases/security/encryption/choose-an-encryption-algorithm>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

7.3 Ensure Database Backups are Encrypted (Automated)

Profile Applicability:

- Level 2 - Database Engine

Description:

Ensure Database Backups are Encrypted

Rationale:

Databases may contain sensitive data. Backups of this data allow the data to easily leave the Enterprise and secure environments. Encrypting the backup makes accessing the data much more difficult.

Impact:

A database backup accidentally exposed to the Internet or transmitted outside a secure environment can be easily restored to a SQL Server anywhere and its contents discovered.

Audit:

```
SELECT
b.key_algorithm, b.encryptor_type, d.is_encrypted,
    b.database_name,
    b.server_name,
    b.backup_finish_date
FROM msdb.dbo.backupset b
inner join sys.databases d on b.database_name = d.name
where b.key_algorithm IS NULL AND b.encryptor_type IS NULL AND d.is_encrypted
= 0;
```

No rows should be returned to be compliant.

Remediation:

SQL Server backups need to 'Back up to a new media set', not 'Back up to the existing media set' in order to allow for encryption. The backup option to **Encrypt Backup** can be implemented after a Certificate or Asymmetric key has been applied to the SQL Server for this purpose.

Alternatively, encrypt the database with TDE. This automatically encrypts the backups as well. See 7.5

Additional Information:

We recommend you purge backup history prior to the date you began using encryption.

Use **sp_delete_backuphistory** to purge old records.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.11 <u>Encrypt Sensitive Data at Rest</u> Encrypt sensitive data at rest on servers, applications, and databases containing sensitive data. Storage-layer encryption, also known as server-side encryption, meets the minimum requirement of this Safeguard. Additional encryption methods may include application-layer encryption, also known as client-side encryption, where access to the data storage device(s) does not permit access to the plain-text data.		●	●
v8	11.3 <u>Protect Recovery Data</u> Protect recovery data with equivalent controls to the original data. Reference encryption or data separation, based on requirements.	●	●	●
v7	14.8 <u>Encrypt Sensitive Information at Rest</u> Encrypt all sensitive information at rest using a tool that requires a secondary authentication mechanism not integrated into the operating system, in order to access the information.			●

7.4 Ensure Network Encryption is Configured and Enabled (Automated)

Profile Applicability:

- Level 2 - Database Engine

Description:

Configuring and enabling network encryption ensures traffic between the application and the database system is encrypted. This will ensure compliance to security standards such as PCI DSS, which is required if connections are through a public network.

Network encryption can be configured in SQL Server either with self-signed certificates or TLS certificates.

Rationale:

Network encryption will ensure data transmitted over the network is protected, so attackers can't ex-filtrate passwords, and confidential data. This protects against man in the middle attack, and network sniffing attacks to ex-filtrate data transmitted between the database system and applications.

Audit:

Run the following T-SQL code against the Master database:

```
SELECT DISTINCT encrypt_option
FROM sys.dm_exec_connections c
WHERE net_transport <> 'Shared memory'
AND c.endpoint_id NOT IN (
    SELECT endpoint_id
    FROM sys.database_mirroring_endpoints
    WHERE encryption_algorithm IS NOT NULL
);
```

A response of TRUE implies a pass and therefore compliant.

Remediation:

Refer to Microsoft SQL Server Encryption Documentation:

<https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/sql-server-encryption>

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-sql-server-encryption>
2. <https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/sql-server-encryption>
3. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/certificate-requirements>
4. <https://learn.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/sys-dm-exec-connections-transact-sql>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

7.5 Ensure Databases are Encrypted with TDE (Automated)

Profile Applicability:

- Level 2 - Database Engine

Description:

Ensure user databases are encrypted using Transparent Data Encryption (TDE). Backups of databases encrypted with TDE are automatically encrypted as well.

Rationale:

A malicious party who steals physical media like drives or backup tapes can restore or attach the database and browse its data.

One solution is to encrypt sensitive data in a database and use a certificate to protect the keys that encrypt the data. This solution prevents anyone without the keys from using the data.

Impact:

A database datafile, logfile or backup accidentally exposed to the Internet or transmitted outside a secure environment can be easily copied/restored to a SQL Server anywhere and its contents discovered.

Audit:

```
select database_id, name, is_encrypted from sys.databases
where database_id > 4 and is_encrypted != 1
```

No rows should be returned to be compliant.

Remediation:

Implement TDE encryption on each user database with sensitive data.

More info on how to do this is available here: <https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption>

References:

1. <https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption>
2. <https://learn.microsoft.com/en-us/archive/blogs/sqlsecurity/feature-spotlight-transparent-data-encryption-tde>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.11 <u>Encrypt Sensitive Data at Rest</u> Encrypt sensitive data at rest on servers, applications, and databases containing sensitive data. Storage-layer encryption, also known as server-side encryption, meets the minimum requirement of this Safeguard. Additional encryption methods may include application-layer encryption, also known as client-side encryption, where access to the data storage device(s) does not permit access to the plain-text data.			
v7	14.8 <u>Encrypt Sensitive Information at Rest</u> Encrypt all sensitive information at rest using a tool that requires a secondary authentication mechanism not integrated into the operating system, in order to access the information.			

8 Appendix: Additional Considerations

This appendix discusses possible configuration options for which no recommendation is being given.

8.1 Ensure 'SQL Server Browser Service' is configured correctly (Manual)

Profile Applicability:

- Level 1 - Database Engine

Description:

No recommendation is being given on disabling the SQL Server Browser service.

Rationale:

In the case of a default instance installation, the SQL Server Browser service is disabled by default. Unless there is a named instance on the same server, there is typically no reason for the SQL Server Browser service to be running. In this case it is strongly suggested that the SQL Server Browser service remain disabled.

When it comes to named instances, given that a security scan can fingerprint a SQL Server listening on any port, it's therefore of limited benefit to disable the SQL Server Browser service.

However, if all connections against the named instance are via applications and are not visible to end users, then configuring the named instance to listening on a static port, disabling the SQL Server Browser service, and configuring the apps to connect to the specified port should be the direction taken. This follows the general practice of reducing the surface area, especially for an unneeded feature.

On the other hand, if end users are directly connecting to databases on the instance, then typically having them use *ServerName\InstanceName* is best. This requires the SQL Server Browser service to be running. Disabling the SQL Server Browser service would mean the end users would have to remember port numbers for the instances. When they don't that will generate service calls to IT staff. Given the limited benefit of disabling the service, the trade-off is probably not worth it, meaning it makes more business sense to leave the SQL Server Browser service enabled.

Audit:

Check the SQL Browser service's status via *services.msc* or similar methods.

Remediation:

Enable or disable the service as needed for your environment.

Default Value:

The SQL Server Browser service is disabled if only a default instance is installed on the server. If a named instance is installed, the default value is for the SQL Server Browser service to be configured as Automatic for startup.

References:

1. <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/sql-server-browser-service-database-engine-and-ssas>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.4 <u>Implement and Manage a Firewall on Servers</u> Implement and manage a firewall on servers, where supported. Example implementations include a virtual firewall, operating system firewall, or a third-party firewall agent.			
v8	4.5 <u>Implement and Manage a Firewall on End-User Devices</u> Implement and manage a host-based firewall or port-filtering tool on end-user devices, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.			
v7	9.2 <u>Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

9 Appendix - Establishing an Audit/Scan User

The following script gives enough permissions to run Audit Procedure scripts, and removes the need to create a user in all databases for the **cis-scan** login/user:

No **sysadmin** or **db_owner** permissions needed.

```
USE [master]
GO

IF not exists (select * from sys.server_principals where name = 'DOMAIN\cis-scan') CREATE LOGIN [DOMAIN\cis-scan]
FROM WINDOWS WITH DEFAULT_DATABASE=[master]
GO

ALTER SERVER ROLE [##MS_DatabaseConnector##] ADD MEMBER [DOMAIN\cis-scan]
GO

ALTER SERVER ROLE [##MS_ServerStateReader##] ADD MEMBER [DOMAIN\cis-scan]
GO

ALTER SERVER ROLE [##MS_SecurityDefinitionReader##] ADD MEMBER [DOMAIN\cis-scan]
GO

USE msdb
GO

IF not exists (select * from sys.database_principals where name = 'cis-scan')
CREATE USER [cis-scan] for login [DOMAIN\cis-scan]
GO

GRANT SELECT ON dbo.sysproxies to [cis-scan]
GRANT SELECT ON dbo.sysproxylogin to [cis-scan]
GO
```

Appendix: Summary Table

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
1	Installation, Updates and Patches		
1.1	Ensure Latest SQL Server Cumulative and Security Updates are Installed (Manual)	☐	☐
1.2	Ensure Single-Function Member Servers are Used (Manual)	☐	☐
2	Surface Area Reduction		
2.1	Ensure 'Ad Hoc Distributed Queries' Server Configuration Option is set to '0' (Automated)	☐	☐
2.2	Ensure 'CLR Enabled' Server Configuration Option is set to '0' (Automated)	☐	☐
2.3	Ensure 'Cross DB Ownership Chaining' Server Configuration Option is set to '0' (Automated)	☐	☐
2.4	Ensure 'Database Mail XPs' Server Configuration Option is set to '0' (Automated)	☐	☐
2.5	Ensure 'Ole Automation Procedures' Server Configuration Option is set to '0' (Automated)	☐	☐
2.6	Ensure 'Remote Access' Server Configuration Option is set to '0' (Automated)	☐	☐
2.7	Ensure 'Remote Admin Connections' Server Configuration Option is set to '0' (Automated)	☐	☐
2.8	Ensure 'Scan For Startup Procs' Server Configuration Option is set to '0' (Automated)	☐	☐
2.9	Ensure 'Trustworthy' Database Property is set to 'Off' (Automated)	☐	☐

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
2.10	Ensure Unnecessary SQL Server Protocols are set to 'Disabled' (Manual)	☐	☐
2.11	Ensure SQL Server is configured to use non-standard ports (Automated)	☐	☐
2.12	Ensure 'Hide Instance' option is set to 'Yes' for Production SQL Server instances (Automated)	☐	☐
2.13	Ensure the 'sa' Login Account is set to 'Disabled' (Automated)	☐	☐
2.14	Ensure the 'sa' Login Account has been renamed (Automated)	☐	☐
2.15	Ensure 'AUTO_CLOSE' is set to 'OFF' on contained databases (Automated)	☐	☐
2.16	Ensure no login exists with the name 'sa' (Automated)	☐	☐
2.17	Ensure 'clr strict security' Server Configuration Option is set to '1' (Automated)	☐	☐
3	Authentication and Authorization		
3.1	Ensure 'Server Authentication' Property is set to 'Windows Authentication Mode' (Automated)	☐	☐
3.2	Ensure CONNECT permissions on the 'guest' user is Revoked within all SQL Server databases (Automated)	☐	☐
3.3	Ensure 'Orphaned Users' are Dropped From SQL Server Databases (Automated)	☐	☐
3.4	Ensure SQL Authentication is not used in contained databases (Automated)	☐	☐
3.5	Ensure the SQL Server's MSSQL Service Account is Not an Administrator (Manual)	☐	☐
3.6	Ensure the SQL Server's SQLAgent Service Account is Not an Administrator (Manual)	☐	☐

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
3.7	Ensure the SQL Server's Full-Text Service Account is Not an Administrator (Manual)	☐	☐
3.8	Ensure only the default permissions specified by Microsoft are granted to the public server role (Automated)	☐	☐
3.9	Ensure Windows BUILTIN groups are not SQL Logins (Automated)	☐	☐
3.10	Ensure Windows local groups are not SQL Logins (Automated)	☐	☐
3.11	Ensure the public role in the msdb database is not granted access to SQL Agent proxies (Automated)	☐	☐
3.12	Ensure the 'SYSADMIN' Role is Limited to Administrative or Built-in Accounts (Manual)	☐	☐
3.13	Ensure no admin role membership in MSDB database (Automated)	☐	☐
3.14	Ensure 'Control Server' permission is not granted (Manual)	☐	☐
4	Password Policies		
4.1	Ensure 'MUST_CHANGE' Option is set to 'ON' for All SQL Authenticated Logins (Manual)	☐	☐
4.2	Ensure 'CHECK_EXPIRATION' Option is set to 'ON' for All SQL Authenticated Logins Within the Sysadmin Role (Automated)	☐	☐
4.3	Ensure 'CHECK_POLICY' Option is set to 'ON' for All SQL Authenticated Logins (Automated)	☐	☐
5	Auditing and Logging		
5.1	Ensure 'Maximum number of error log files' is set to greater than or equal to '12' (Automated)	☐	☐

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
5.2	Ensure 'Default Trace Enabled' Server Configuration Option is set to '1' (Automated)	☐	☐
5.3	Ensure 'Login Auditing' is set to 'failed logins' (Automated)	☐	☐
5.4	Ensure 'SQL Server Audit' is set to capture both 'failed' and 'successful logins' (Automated)	☐	☐
6	Application Development		
6.1	Ensure Database and Application User Input is Sanitized (Manual)	☐	☐
6.2	Ensure 'CLR Assembly Permission Set' is set to 'SAFE_ACCESS' for All CLR Assemblies (Automated)	☐	☐
7	Encryption		
7.1	Ensure 'Symmetric Key encryption algorithm' is set to 'AES_128' or higher in non-system databases (Automated)	☐	☐
7.2	Ensure Asymmetric Key Size is set to 'greater than or equal to 2048' in non-system databases (Automated)	☐	☐
7.3	Ensure Database Backups are Encrypted (Automated)	☐	☐
7.4	Ensure Network Encryption is Configured and Enabled (Automated)	☐	☐
7.5	Ensure Databases are Encrypted with TDE (Automated)	☐	☐
8	Appendix: Additional Considerations		
8.1	Ensure 'SQL Server Browser Service' is configured correctly (Manual)	☐	☐
9	Appendix - Establishing an Audit/Scan User		

Appendix: Change History

Date	Version	Changes for this version
Mar 24, 2023	1.1.0	2.12 Ensure 'Hide Instance' option is set to 'Yes' for Production SQL Server instances (Ticket 18089)
Sep 21, 2023	1.1.0	UPDATE - 2.11 Artifact (Ticket 17832)
Oct 5, 2023	1.1.0	Remove references to cluster (Ticket 19944)
Oct 5, 2023	1.1.0	Updated code, copy from 2019 Benchmark (Ticket 19936)
Oct 19, 2023	1.1.0	section 7.4 Ensure Network Encryption is Configured and Enabled is added (Ticket 17852)
Oct 19, 2023	1.1.0	Add new recommendation 7.4 (Ticket 20110)
Oct 19, 2023	1.1.0	Add new recommendation 7.5 (Ticket 20111)
Nov 2, 2023	1.1.0	CIS Benchmarks and Assessor Scanning Tool for Microsoft SQL Server 2022 (Ticket 17250)
Nov 16, 2023	1.1.0	Remediation procedure by TSQL command (Ticket 17918)
Nov 16, 2023	1.1.0	Audit procedure by TSQL command (Ticket 17919)
Nov 16, 2023	1.1.0	New code suggestion + discussion (Ticket 19858)

Date	Version	Changes for this version
Jan 11, 2024	1.1.0	3.1 is not practical (Ticket 17862)
Jan 25, 2024	1.1.0	Review T-SQL for 2.11 (Ticket 20341)
Jan 25, 2024	1.1.0	section 3.12 Ensure SYSADMIN role is granted strictly to database administrators and to designated built-in Microsoft Accounts is added (Ticket 17855)
Feb 14, 2024	1.1.0	Server 2022 does not have "Ensure Databases are Encrypted with TDE" control like Server 2019 (Ticket 20937)
Feb 16, 2024	1.1.0	Add - 5.5 Ensure 'SQL Server Audit' Captures Server-Level Role Authorization Processes (Ticket 20647)
Mar 7, 2024	1.1.0	ADD - Ensure membership in db_owner role in MSDB database is limited (Ticket 20912)
Mar 7, 2024	1.1.0	Create Appendix - Establishing an Audit/Scan User (Ticket 18983)

Date	Version	Changes for this version
Mar 7, 2024	1.1.0	Oracle CIS Benchmark documentation provided a detailed Appendix for Establishing an Audit/Scan User - providing the appropriate permissions to be used to execute the queries and perform other assessment actions. While this could be accomplished by grantin (Ticket 16951)
Mar 14, 2024	1.1.0	DB_OWNER ROLE COMPLIANCE IN MSDB database (Ticket 16291)
Apr 23, 2024	1.2.0	Item 2.11 - TSQL audit statement correction (Ticket 21477)
May 2, 2024	1.2.0	3.13 T-SQL Query is wrong (Ticket 21547)
May 2, 2024	1.2.0	Clarify that "no limit" is also acceptable (Ticket 21630)
May 7, 2024	1.2.0	SQL will never find a result for 3.3 (Ticket 21670)
Jul 18, 2024	1.2.0	SQL Server Database ciscat report collection using powershell (Ticket 22069)
Jul 18, 2024	1.2.0	email feedback@cisecurity.org not working (Ticket 21816)
Jul 18, 2024	1.2.0	Reference in Asymmetric Key section (7.2) of SQL Server 2022 1.1.0 (Ticket 21815)

Date	Version	Changes for this version
Jul 18, 2024	1.2.0	Updated SQL for instances configured with ListenOnAllIPs incorrect (Ticket 21669)
Oct 15, 2024	1.2.0	Query in 3.13 does not work (Ticket 21677)
Oct 15, 2024	1.2.0	Not supported in Express (Ticket 21679)
Oct 15, 2024	1.2.0	Not supported in Express (Ticket 21680)
Feb 10, 2025	1.2.0	Control to detect use of weak cipher suites (Ticket 23652)
Mar 6, 2025	1.2.0	Hardening Standard for Microsoft SQL Server & Database (Ticket 22461)
Apr 30, 2025	1.2.0	SQL Server 2022 (Ticket 24713)
May 2, 2025	1.2.0	SQL Server 2022 Benchmark showing false failures (Ticket 22969)
May 2, 2025	1.2.0	Enhancement for control 3.5 Ensure the SQL Server's MSSQL Service Account is Not an Administrator (Manual) (Ticket 21875)
May 7, 2025	1.2.0	Ensure 'SQL Server Audit' is set to capture both 'failed' and 'successful logins' (Ticket 21893)
May 14, 2025	1.2.0	UPDATE T-SQL for recommendation 4.1 (Ticket 24867)

Date	Version	Changes for this version
May 14, 2025	1.2.0	Check 7.3 applies to system databases. To ensure compliance with 7.3, it is necessary to encrypt both the system databases and their corresponding backup databases. (Ticket 22610)
May 14, 2025	1.2.0	Ensure Network Encryption is Configured and Enabled vulnerability in tenable nessus scan (Ticket 23175)
Jul 10, 2025	1.3.0	SQL Server 2022 CIS (1.1.0) control 6.2 Ensure 'CLR Assembly Permission Set' is set to 'SAFE_ACCESS' for All CLR Assemblies' (Ticket 24525)
Aug 19, 2025	1.3.0	Automation tool to apply CIS benchmarks (Ticket 25738)
Sep 4, 2025	1.3.0	Minor suggestion - change <NumberAbove12> to <NumberAbove11> (Ticket 25883)
Sep 18, 2025	1.3.0	CHECK_EXPIRATION - distributor_admin (Ticket 25976)
Oct 2, 2025	1.3.0	Recommendation 3.13 is executed in wrong database (Ticket 25894)
Oct 2, 2025	1.3.0	The SQL for this new benchmark is incorrect (Ticket 25926)
Oct 16, 2025	1.3.0	Proposed change to Rationale Statement for 2.11 (Ticket 26238)

Date	Version	Changes for this version
Nov 13, 2025	1.3.0	USER type - missing ROLE in query (Ticket 26535)
Mar 5, 2026	1.3.0	Simplifying this script using new server roles (Ticket 27371)
Apr 1, 2026	1.3.0	Update Audit Procedure T-SQL (Ticket 27528)
Apr 1, 2026	1.3.0	Bug in Orphaned users check - even with no orphaned users, still shows as failed (Ticket 26663)
Apr 1, 2026	1.3.0	UPDATE - 3.5 Ensure the SQL Server's MSSQL Service Account is Not an Administrator (Ticket 27529)
Apr 1, 2026	1.3.0	UPDATE - 3.6 Ensure the SQL Server's SQLAgent Service Account is Not an Administrator (Ticket 27530)
Apr 8, 2026	1.3.0	Update Audit procedure text (Ticket 27607)
Apr 8, 2026	1.3.0	proposing new recommendation: 3.14 Ensure 'control server' permission is not granted (Ticket 25257)
Apr 8, 2026	1.3.0	Excluding the principal in the query (Ticket 27608)
Apr 30, 2026	1.3.0	Inquiry Regarding CIS Level 2 Benchmark 3.13 Failure due to Azure Arc SQL Extension Role (Ticket 26426)

Date	Version	Changes for this version
May 14, 2026	1.3.0	xp_cmdshell attacks control recommendation (Ticket 22609)
May 14, 2026	1.3.0	It seems that the MSSQL2022 security CIS benchmark does not have the check which ensures xp_cmdshell server configuration option is set to 0 aka disabled. Please can we have this control added in this benchmark. (Ticket 22926)
May 14, 2026	1.3.0	Even after enabling audit in SQL 2022, scan report is showing as failed for this recommendation. (Ticket 27234)
May 14, 2026	1.3.0	Audit procedure does not account for disabled accounts (Ticket 27794)
May 14, 2026	1.3.0	UPDATE text in recommendation (Ticket 27797)
May 14, 2026	1.3.0	UPDATE Audit Procedure (Ticket 27796)
May 19, 2026	1.3.0	4 Typos in the Change History section of the PDF (Ticket 27832)
May 19, 2026	1.3.0	Add missing word in Rationale Statement (Ticket 27831)
May 19, 2026	1.3.0	Replace one word in Impact Statement (Ticket 27830)

Date	Version	Changes for this version
May 19, 2026	1.3.0	SQL to remediate has an error (Ticket 27825)
May 19, 2026	1.3.0	Error in Remediation Procedure SQL (Ticket 27827)
May 19, 2026	1.3.0	First sentence in Rationale Statement needs improvement (Ticket 27828)
May 19, 2026	1.3.0	SQL Server Common Criteria (Ticket 27799)
May 19, 2026	1.3.0	Two sentences can be improved (Ticket 27829)
May 19, 2026	1.3.0	An external URL in the References is incorrect (Ticket 27826)