# Table of contents

1. Introduction .......................................................................................................................... 3  
2. Pre-installation ...................................................................................................................... 4  
   Log-on and service accounts .......................................................................................... 4  
   File system ..................................................................................................................... 5  
   Network connectivity ...................................................................................................... 6  
   Security updates ............................................................................................................. 7  
3. SQL Server services and connections .................................................................................. 9  
4. Post-installation – reducing surface area .......................................................................... 10  
5. Controlling access ........................................................................................................... 17  
6. Segregate support staff logins ........................................................................................... 21  
7. Secure system objects and catalog .................................................................................... 25  
8. Secure user objects using schemas .................................................................................... 26  
9. On-going password maintenance ....................................................................................... 27  
10. Data transmission encryption – database engine .............................................................. 28  
11. Transparent database encryption – database engine ....................................................... 31  
12. Auditing access – database engine .................................................................................... 38
1. Introduction

Securing SQL Server can be viewed as a series of steps involving four stages: pre-installation, installation, post-installation, and features enablement. This document provides security hardening Reference for customer for managing and implementing SQL Server environments, in particular, the database engine component. The rest of the document dives into detail in each of the areas.
2. Pre-installation

Log-on and service accounts

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| 2.1. | Accounts segregation of duties | Separate service accounts to minimize opportunities for internal threads and fraud | Each SQL Server service should be using different service account  
• SQL Server (database engine)  
• SQL Server Agent  
• SQL Server Browser  
• SQL Server Analysis Services  
• SQL Server Reporting Services  
• SQL Server Integration Services  
• SQL Server Full-text Search  
• SQL Server VSS writer | |
| 2.2. | Service account – disallow interactive logon | SQL Server service accounts does not require interactive logon privilege | Do not grant “interactive logon” to any of the SQL Server service accounts | |
| 2.3. | Service account – not in local administrators group | SQL Server service accounts does not require local administrative privileges | Do not assign any of the SQL Server service accounts to the “local administrators” group | |
| 2.4. | Installation logon account – allow interactive login | SQL Server installation account requires interactive logon privilege | Grant “interactive logon” to the installation account when installing SQL Server | |
| 2.5. | Installation logon account – member of local administrators group | SQL Server installation account does require local administrative privileges | Ensure the installation account used for installation SQL Server possess local administrative privilege | |
| 2.6. | Grant “allow log on locally” privileges to system admin only. | DBA should manager SQL Server instance and database level only, and is not allowed to access Operation System level. | Grant “allow log on locally” privileges to system admin only. | |
| 2.7. | Service account provisioning – SQL Server database engine | Only assign the minimally-required privilege.  
NOTE – SQL Server setup automatically grants these permissions to the per-service SID running on Windows Server 2008 and onwards:  
Default Instance: SQLServerMSSQLUser$ComputerName$MSSQLSERVER  
Named instance: SQLServerMSSQLUser$ComputerName$InstanceName | • Log on as a service  
• Replace a process-level token  
• Bypass traverse checking  
• Adjust memory quotas for a process  
• Permission to start SQL Writer  
• Permission to read the Event Log service  
• Permission to read the Remote Procedure Call service | |

| 2.8. | Service account provisioning – SQL Server Agent | Only assign the minimally-required privilege. NOTE – SQL Server setup automatically grants these permissions to the per-service SID. | • Log on as a service  
• Replace a process-level token  
• Bypass traverse checking  
| 2.9. | Service account provisioning – Integration Services | Only assign the minimally-required privilege. NOTE – SQL Server setup automatically grants these permissions to the per-service SID. | • Log on as a service  
• Bypass traverse checking  
• Permission to write to application event log  
• Impersonate a client after authentication  
| 2.10. | Service account provisioning – Full-text Search | Only assign the minimally-required privilege. NOTE – SQL Server setup automatically grants these permissions to the per-service SID. | • Log on as a service | [https://msdn.microsoft.com/en-us/library/ms143504(v=sql.110).aspx](https://msdn.microsoft.com/en-us/library/ms143504(v=sql.110).aspx) |
| 2.12. | Service account provisioning – SQL Server VSS Writer | N/A | The SQLWriter service runs under the LOCAL SYSTEM account which has all the required permissions. SQL Server setup does not check or grant permissions for this service. | [https://msdn.microsoft.com/en-us/library/ms143504(v=sql.110).aspx](https://msdn.microsoft.com/en-us/library/ms143504(v=sql.110).aspx) |

**File system**

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>


2.13. Configure a secure file system

NTFS file system should be chosen over FAT as it enables security options like file and directory access control lists (ACLs) and encrypting file system (EFS) file encryption. During installation, SQL Server will set appropriate ACLs on registry keys and files if it detects NTFS.

Optional – can consider enabling EFS or BitLocker onto drives used by SQL Server for additional security measure, at the expense of performance and supportability.

### Network connectivity

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.14.</td>
<td>Firewall requirements</td>
<td>On Windows Server 2008 and later, Windows firewall is enabled and is blocking remote connections</td>
<td>Allow for incoming connection to SQL Server through the specified SQL client ports, as well as allowing other communication ports.</td>
<td>More on SQL connectivity port later in this document.</td>
</tr>
<tr>
<td>2.15.</td>
<td>Port requirements – database engine client connectivity</td>
<td>Database engine requires a number of ports by default in order to operate. It differs between default instance and named instances.</td>
<td>Default instance – defaults to TCP 1433, should change to a less-common port. Named instance – defaults to dynamic port above 5000. Should change to a less-common fixed port.</td>
<td>More on SQL connectivity port later in this document.</td>
</tr>
<tr>
<td>2.16.</td>
<td>Port requirements – SQL Server Browser service</td>
<td>The SQL Server Browser service listens for incoming connections to a named instance and provides the client the TCP port number that corresponds to that named instance. Normally the SQL Server Browser service is started whenever named instances of the Database Engine are used. The SQL Server Browser service does not have to be started if the client is configured to connect to the specific port of the named instance.</td>
<td>Defaults to UDP 1434. If service is disabled, client must make connection to SQL Server using pre-defined port.</td>
<td>More on SQL Server browser service later in this document.</td>
</tr>
<tr>
<td>2.17.</td>
<td>Port requirements – for SQL Server Configuration Manager</td>
<td>SQL Server Configuration Manager uses WMI to list and manage services.</td>
<td>Defaults to TCP 135. WMI runs as part of a shared service host with ports assigned through DCOM.</td>
<td></td>
</tr>
<tr>
<td>2.18.</td>
<td>Port requirements – Microsoft Distributed</td>
<td>If the connecting application uses distributed transactions, the administrator have to configure the firewall to allow Microsoft</td>
<td>Defaults to TCP 135.</td>
<td></td>
</tr>
</tbody>
</table>
Transaction Coordinator (MS DTC) | Distributed Transaction Coordinator (MS DTC) traffic to flow between separate MS DTC instances, and between the MS DTC and resource managers such as SQL Server. | If used, suggest using the pre-configured “Distributed Transaction Coordinator” rule group in Windows Firewall. | 

2.19. Port requirements – dedicated admin connection (DAC) | Allows an administrator to connect to a server to execute diagnostic functions or Transact-SQL statements. The DAC can be used when the Database Engine will not respond to regular connections. | Default instance – defaults to TCP 1434. Named instance – defaults to a random TCP port above 5000. More on dedicated admin connection later in this document. | Remote DAC is disabled by default. Once enabled, incoming ports must be opened to allow for such connection. | 

2.20. Port requirements – SQL Server running on Windows Server Failover Cluster (WSFC) | WSFC uses a number of ports for inter-communication. | Defaults to the following: • 135 (RPC endpoint mapper/DCOM) • 135 (RPC endpoint mapper over UDP) • 3343 (used by the Cluster Network Driver) • 445 (SMB) • 139 (NetBIOS session service) | 

2.21. Port requirements – SQL Server service running over HTTP endpoint | Endpoints are essentially web services that expose database access over HTTP, allowing communication into SQL Server via HTTP endpoint. | No default port but conventional configuration is to use TCP 80 | No default endpoint / port is configured. | 

2.22. Port requirements – SQL Server service running | Endpoints are essentially web services that expose database access over HTTPS, allowing communication into SQL Server using SSL. | No default port but conventional configuration is to use TCP 443 | No default endpoint / port is configured. | 

2.23. Port requirements – Service Broker service | Enables SQL Server Service Broker service (SSB) communications. | No default port but conventional configuration is to use TCP 4022 | No default endpoint / port is configured. | 

2.24. Port requirements – Database mirroring/ AlwaysOn AG | Allows SQL Server to use TCP endpoint for database mirroring configurations. | No default port but conventional configuration is to use TCP 5022 or 7022 | No default endpoint / port is configured. | 

Security updates

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.25</td>
<td>Ensure server OS and SQL are current with latest security hotfixes and service packs</td>
<td>Regularly apply security-related hotfixes to SQL Server installation prevents from known attacks.</td>
<td>Follow your organization’s practice – either through centrally-managed server such as WSUS or Microsoft Update, or manually update server with the latest security hotfixes and service packs.</td>
<td></td>
</tr>
</tbody>
</table>
## Installation

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.26.</td>
<td>Install minimal and only required components</td>
<td>Reduce surface area at installation time</td>
<td>Only install components that will be needed immediately. Additional components, if required, can always be installed at a later stage</td>
<td></td>
</tr>
</tbody>
</table>
| 2.27. | Separate drives for binaries, data / log, and backup | Separate partitions provide greater protections via host and file permissions at the volume level as well as allowing greater control over data storage usage and monitoring of the database. | During installation time, choose separate drives for SQL binaries, data, log, and backup files. If data, log or backup locations need to be changed after installation, it can be done via Management Studio:  
• Right-click on server instance, choose the “Facet” option  
• Under the Facet dropdown, choose “Server Settings”  
• Modify the value of “BackupDirectory”, “DefaultFile” and “DefaultLog” locations | |
| 2.28. | Windows Authentication | Using Windows logins with SQL Server enables single sign-on and simplifies login administration | During installation, choose the Windows authentication model.  
If at a later stage this needs to change to Mixed-mode authentication, it can be done via Management Studio:  
• Right-click and Properties on the Server instance  
• Under the Security page, select Mixed mode authentication mode check box  
NOTE - Restart SQL Server for it to take effect. | Default installation option is “Windows Authentication” only |
### 3. SQL Server services and connections

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.</td>
<td>Database Engine service</td>
<td>Startup can be set to Automatic or Manual during installation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2.</td>
<td>Database Engine remote</td>
<td>Remote connections are disabled in SQL Server</td>
<td><strong>Remote connections are disabled in SQL Server Express, Evaluation, and Developer editions</strong> and are enabled in other editions.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>connections</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3.</td>
<td>SQL Server Agent service</td>
<td>If installed, the startup type is set to Manual and the service is stopped.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4.</td>
<td>Full-text Search service</td>
<td>If installed, startup is set to Manual and the service is stopped.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5.</td>
<td>Integration Services service</td>
<td>If installed, the service is set to the Automatic startup mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6.</td>
<td>SQL Server Browser service</td>
<td>Startup is set to Automatic in the following conditions:</td>
<td><strong>Startup is set to Automatic in the following conditions:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When there are named instances of the Database Engine or Analysis Services on the server</td>
<td><strong>• When there are named instances of the Database Engine or Analysis Services on the server</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When installed on a cluster</td>
<td><strong>• When installed on a cluster</strong></td>
<td></td>
</tr>
</tbody>
</table>
## 4. Post-installation – reducing surface area

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| 4.1. | SQL connectivity protocols | Remove unnecessary connectivity protocols | Disable "named pipes" and enable only "TCP/IP" and "shared memory" for both server and client components. In SQL Server Configuration Manager:  
Server component:  
- Left pane, SQL Server Network Configuration  
- Under Protocols for <instance>  
- Right pane, disable “named pipes”, leaving both “TCP/IP” and “Shared memory” enabled.  
Client component:  
- Left pane, SQL Native Client 11.0 Configuration  
- Under Client Protocols  
- Right pane, disable “named pipes”, leaving both “TCP/IP” and “Shared memory” enabled  
- Repeat for SQL Native Client 11.0 Configuration (32bit), if applicable | Default value is enabled. VIA is removed from SQL 2012. |

| 4.2. | SQL Server client port (TCP) | Use non-default client port helps protect from common attacks against well-known ports | For both default and named SQL instances, change from the connecting client port from default:  
Server component:  
- Left pane, SQL Server Network Configuration  
- Under Protocols for <instance>  
- Right pane, right-click and Properties on “TCP/IP”  
- Under “IP Addresses” tab, under IPALL change TCP port to a non-default port  
- Restart SQL Server to take effect  
Connectivity scenarios when using non-default port:  
SQL Browser service enabled:  
- Client, without specifying which SQL port to use, connects to server via 1434 initially | Default instance uses 1433 whilst named instance uses a dynamic port above 5000. |
| 4.3. | Remove sample databases | Unnecessary sample database files installed could potentially expose the production environment, by providing a vulnerability or a mechanism through which an individual may gain unauthorized or inappropriate access to the system. | Ensure the following known sample databases are removed from production systems:
- Pubs
- Northwind
- AdventureWorks* (may have suffix of 2012, LT, DW, or a combination of it) | SQL Browser service disabled:
- Client, must specify the protocol, IP / network name / FQDN, as well as the listening port in order to connect
- No other method to resolve the non-default listening port | SQL Server 2012 installation does not install any sample databases |

| 4.4. | Blocked process threshold | Blocked Process threshold can be set from 0 to 86,400 at which blocked process reports are generated. | Adjust the blocked process threshold to specify the threshold to 20 seconds – this can be done via Management Studio:
- Right-click on server instance, choose the “Facet” option
- Under the Facet dropdown, choose “Server Configuration”
- Modify the value of “BlockedProcessThreshold” | Default value is 0 |

| 4.5. | Cross-database chaining | Use the cross database ownership chaining option to configure cross-database ownership chaining for an instance of SQL Server. This server option allows you to control cross-database ownership chaining at the database level or to allow cross-database ownership chaining for all databases. Has two options – server-level (all databases); or per database-level | Disabled
Only consider enabling this option when fully testing application functionality and understanding possible security risks associated with it. If required – this can be done via Management Studio:
- Right-click on server instance, choose the “Facet” option
- Under the Facet dropdown, choose “Server Configuration” | Default is false on both server-level and database-level |
<table>
<thead>
<tr>
<th>4.6.</th>
<th>Default trace enabled</th>
<th>Default trace provides troubleshooting assistance to database administrators by ensuring that they have the log data necessary to diagnose problems the first time they occur</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>NOTE – requires to execute using “sysadmin” server-role member</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default is True</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.7.</td>
<td>Ad hoc remote distributed queries</td>
<td>Ad hoc distributed queries use the OPENROWSET and OPENDATASOURCE functions to connect to remote data sources that use OLE DB. OPENROWSET and OPENDATASOURCE should be used only to reference OLE DB data sources that are accessed infrequently.</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
|       |                        | Can be changed in Management Studio:  
- Right-click on server instance, choose the “Facet” option  
- Under the Facet dropdown, choose “Server Configuration”  
- Examine the value of “AdHocRemoteQueriesEnabled” |
|       |                        | NOTE – requires to execute using “sysadmin” server-role member                                                                                                                                      | Disabled in new installations. |
| 4.8.  | Common language runtime (CLR) integration | Allows the use of CLR-based user-defined functions, procedures, triggers, and types from within the Database Engine | Disabled |
|       |                        | Can be changed in Management Studio:  
- Right-click on server instance, choose the “Facet” option  
- Under the Facet dropdown, choose “Server Configuration”  
- Examine the value of “ClrIntegrationEnabled” |
|       |                        | NOTE – requires to execute using “sysadmin” server-role member                                                                                                                                      | Disabled in new installations. |
| 4.9.  | Database Mail | Allows for sending e-mail messages from the Database Engine using SMTP | Disabled |
|       |                        |                                                                                                                                            | Disabled in new installations. |
| **4.10.** OLE Automation stored procedures | The OLE automation extended stored procedures allow Transact-SQL batches, stored procedures, and triggers to reference custom OLE Automation, such as ones below:  
- `sp_OACreate`  
- `sp_OADestroy`  
- `sp_OAGetErrorInfo`  
- `sp_OAGetProperty`  
- `sp_OAMethod`  
- `sp_OASetProperty`  
- `sp_OAStop` | Disabled  
Can be changed in Management Studio:  
- Right-click on server instance, choose the “Facet” option  
- Under the Facet dropdown, choose “Server Configuration”  
- Examine the value of “OleAutomationEnabled”  
NOTE – requires to execute using “sysadmin” server-role member | Disabled in new installations. |
| **4.11.** Remote Dedicated administrator connection (DAC) | Allows an administrator to connect to a server to execute diagnostic functions or Transact-SQL statements. The DAC can be used when the Database Engine will not respond to regular connections. | Enabled  
To allow the use of DAC from a remote computer.  
Can be changed in Management Studio:  
- Right-click on server instance, choose the “Facet” option  
- Under the Facet dropdown, choose “Server Configuration”  
- Modify the value of “RemoteDacEnabled”  
NOTE – requires to execute using “sysadmin” server-role member  
NOTE2 - Restart SQL Server for it to take effect. | Default is disabled  
(allow for local DAC only) |
| **4.12.** Assign static port for dedicated administrator connection (DAC) | Should configure fixed port for easier network management. | To configure a static port for DAC, registry key that corresponds to the SQL instance must be modified manually. For example, the registry key may be the following:  
Default instance – defaults to TCP 1434.  
Named instance – defaults to a random TCP port above 5000. |
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4.13.</td>
<td>Service Broker Endpoint Active</td>
<td>SQL Server uses Service Broker endpoint for Service Broker communication outside the instance of SQL Server</td>
</tr>
</tbody>
</table>
|   |   |   | Can be changed in Management Studio:  
|   |   |   | • Right-click on server instance, choose the “Facet” option  
|   |   |   | • Under the Facet dropdown, choose “Surface Area Configuration”  
|   |   |   | • Examine the value of “ServiceBrokerEndpointActive”  
|   |   |   | NOTE – requires to execute using “sysadmin” server-role member |
|   |   |   | Default is False - Endpoints are not configured by default. |

4.14. SOAP Endpoint Enabled  

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SOAP endpoint responds to SOAP requests.</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
|   |   | Can be changed in Management Studio:  
|   |   | • Right-click on server instance, choose the “Facet” option  
|   |   | • Under the Facet dropdown, choose “Surface Area Configuration”  
|   |   | • Examine the value of “SoapEndpointsEnabled”  
|   |   | NOTE – requires to execute using “sysadmin” server-role member |
|   |   | Disabled in new installations. |

4.15. SQL Mail  

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Similar to DB Mail, SQL Mail allows for sending e-mail messages from the Database Engine. NOTE – this is deprecated and is for backward-compatibility purpose only</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
|   |   | Can be changed in Management Studio:  
|   |   | • Right-click on server instance, choose the “Facet” option  
|   |   | Disabled in new installations. |

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL.x\MSSQLServer\SuperSocketNetLib\AdminConnection\Tcp  

Value: TcpDynamicPorts  

NOTE – The “x” in “MSSQL.x” is a number that indicates the directory where the instance is installed for SQL Server 2008.  

NOTE2 - Restart SQL Server for it to take effect.
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.16.</td>
<td><code>xp_cmdshell</code></td>
<td>The <code>xp_cmdshell</code> extended stored procedure executes a command string as an operating-system command shell and returns any output as rows of text</td>
<td>Disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.17.</td>
<td>Remote access</td>
<td>Remote access allow trusted and privileged access to the database server from a remote server</td>
<td>Trusted and privileged access to the database server from a remote server must not be permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.18.</td>
<td>Linked server connectivity</td>
<td>Linked server enables SQL Server to execute commands against OLE DB data sources on remote servers</td>
<td>Use linked servers rather than remote servers where required, and configure linked servers to use Windows authentication where required by selecting “Be made using the login’s current security context” in the Security Page of a Linked Server Properties</td>
</tr>
<tr>
<td>4.19.</td>
<td>SQL Server Agent trace message</td>
<td>This is a defense-in-depth measure to reduce threats of a disk exhaustion-based denial of service.</td>
<td>Disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To change using Management Studio:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Choose SQL Server instance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Expand to SQL Server Agent branch</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Right-click and Properties</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• On the General Tab, under the section “Error Log”, uncheck the box “Include execution trace messages”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOTE – SQL Server Agent service must be running in order to view the above option</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Default is disabled</td>
</tr>
</tbody>
</table>

| 4.20. | SQL Server Agent event forwarding | Allows for SQL Server Agent events to be forwarded to another server | Disabled. |
|       |                               |                                                                                       | To change using Management Studio: |
|       |                               |                                                                                       | • Choose SQL Server instance |
|       |                               |                                                                                       | • Expand to SQL Server Agent branch |
|       |                               |                                                                                       | • Right-click and Properties |
|       |                               |                                                                                       | • On the Advanced Tab, under the section “SQL Server event forwarding”, uncheck the box “forward events to a different server” |
|       |                               |                                                                                       | NOTE – SQL Server Agent service must be running in order to view the above option |
|       |                               |                                                                                       | Default is disabled |
### 5. Controlling access

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| 5.1. | Authentication mode | Using Windows logins with SQL Server achieves single sign-on and simplifies login administration | Windows authentication  
In Object Explorer, right-click a server, and then select Properties - select the Security page, select Windows authentication mode check box | Defaults to the value chosen during installation |
| 5.2. | Remove BUILTIN\administrators group | BUILTIN\administrators group directly maps to the OS's local administrators group. Previous versions of SQL Server has this group automatically added into the “sysadmin” server-role | If present, remove this group and if necessary, manually add logins to server-roles.  
NOTE – requires to execute using “sysadmin” server-role member | By default this group does not exist on new SQL Server 2012 installations and is only applicable to upgraded-instances only |
| 5.3. | Alternative login with “sysadmin” server-role | Assign alternative “sysadmin” privilege other than ‘sa’ | Create separate login (Windows or SQL) that is assigned to the “sysadmin” server-role. Keep access to this login minimal.  
NOTE – requires to execute using “sysadmin” server-role member | |
| 5.4. | Disable ‘sa’ login if not being used | Disable ‘sa’ once the alternative “sysadmin” login is created, to prevent unintended access via this privileged account | Regardless of whether the authentication mode is Windows or Mixed, the “sa” login if it is not being used. This can be done via Management Studio:  
• Left pane, SQL instance  
• Under Security branch, expand to Logins  
• Right-click and bring up the Properties of “sa”  
• Under “Status” tab, set login to “Disabled”  
NOTE – requires to execute using “sysadmin” server-role member | If Windows authentication is chosen, the “sa” login is disabled automatically. |
| 5.5. | Rename of ‘sa’ to another non-default name | Renaming well-known logins prevent unintended access | This can be done via Management Studio:  
• Left pane, SQL instance  
• Under Security branch, expand to Logins  
• Right-click and choose “Rename” | |
| 5.6. | Enforce SQL login password policies | When creating SQL logins, options are provided to enforce password policies (such as password complexity). The actual policy follows the OS’s password policy, which may be enforced via Domain’s group policies. | Enforced.  
To change this, for each login, bring up the Properties in Management Studio, and choose “Enforce password policy”.  
To verify if there are SQL logins that do not have policy checked / enforced, execute the following script in Management Studio:  
```sql
USE master; SELECT name, is_policy_checked FROM master.sys.sql_logins (NOLOCK) WHERE name NOT LIKE '##%'; GO
```
|  |  | NOTE – requires to execute using “sysadmin” server-role member | Returned value of 0 means it is not enforced. |
| 5.7. | Internal built-in logins | Do not modify / delete internal built-in logins | All built-in logins starting with “##” should be left as-is. |
| 5.8. | Limit serveradmin / sysadmin server-level role assignment | Limit the login assignment to these privileged server-level roles  
• “sysadmin” – members of this role have complete rights over the SQL Server instance – it is the only role that can add others to the sysadmin role. Moreover, this role bypasses all security checks – even in scenario that a login is blocked from doing something, by nature of the login being a member of this role, it will ignore that restriction.  
• “serveradmin” – members of serveradmin can control the SQL Server configuration and even shutdown SQL Server by issuing the SHUTDOWN command if connected.  
As a result, membership in these roles should be carefully controlled – as a minimum, the | Regularly review the server-role assignment and revoke unnecessary logins. To list out all server-role currently assigned, run the following script in Management Studio:  
```sql
USE master; SELECT c.name as Fixed_roleName, a.name as logins ,a.type_desc FROM sys.server_principals a (NOLOCK) INNER JOIN sys.server_role_members b (NOLOCK) ON a.principal_id = b.member_principal_id  
INNER JOIN sys.server_principals c (NOLOCK) ON c.principal_id = b.role_principal_id  
ORDER BY c.name, a.type_desc, a.name; GO
```
|  |  | NOTE – requires to execute using “sysadmin” server-role member |  |
| **5.9.** | **Disable GUEST account in user databases** | **Guest account should not be enabled in any user databases.**

NOTE – master, tempdb or msdb must have GUEST account enabled. | **Disabled**

Suggest running the following script in Management Studio to verify if guest account is indeed “disabled”.

```sql
USE master;
IF NOT EXISTS (SELECT name FROM tempdb.sys.objects WHERE name LIKE '##tblGuestAccess%') CREATE TABLE ##tblGuestAccess (GuestAccess nvarchar(1024));
EXEC sp_MSforeachdb 'USE [?]; INSERT INTO ##tblGuestAccess SELECT ''[?]' + '' >> guest account = '' + cast (hasdbaccess AS CHAR(1)) FROM [?].sys.sysusers (NOLOCK) WHERE [name] = ''guest'';
SELECT * from ##tblGuestAccess;
DROP TABLE ##tblGuestAccess;
GO
```

If found enabled (in user databases only), suggest disabling it using the following script in Management Studio:

```sql
USE <DB_Name>; REVOKE CONNECT FROM guest;
GO
```

**NOTE** – requires to execute using “sysadmin” server-role member |

| **5.10.** | **Limit db_owner database-level role assignment** | **Limit SQL users to db_owner database-role to reduce privileged access** | **Regularly review the database-role assignment and revoke unnecessary SQL users. To list out all users and database-role currently assigned, run the following script in Management Studio:**}
USE master;
IF NOT EXISTS (SELECT name FROM tempdb.sys.objects WHERE name LIKE '##tblDBOwner%') CREATE TABLE ##tblDBOwner (DBname varchar(1024), DBRole varchar(1024), RoleMember varchar(1024), TypeDesc varchar(1024));
EXEC sp_MSforeachdb 'USE [?]; INSERT INTO ##tblDBOwner (DBname, DBRole, RoleMember, TypeDesc) SELECT db_name()as DBNAME, c.name as DB_ROLE ,a.name as Role_Member, a.type_desc FROM sys.database_principals a (NOLOCK) INNER JOIN sys.database_role_members b (NOLOCK) ON a.principal_id = b.member_principal_id INNER JOIN sys.database_principals c (NOLOCK) ON c.principal_id = b.role_principal_id WHERE a.name <> ''dbo'' AND c.is_fixed_role=1 ORDER BY c.name, a.name, a.type_desc;' SELECT * from ##tblDBOwner;
DROP TABLE ##tblDBOwner;
GO

NOTE – requires to execute using “sysadmin” server-role member
6. Segregate support staff logins

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Define access level for read-only support staff login</td>
<td>Purpose of this is to have day-to-day operations staff logging into SQL Server using these “read-only” support staff account. Only in fire-fighting situations then the support staff should login to SQL Server using “sysadmin” server-role logins.</td>
<td>The following lists the privileges the read-only support staff login should and should not have access to: 1. Capture blocking information 2. Execute common DMVs – <code>sys.dm_exec_requests</code>, <code>sys.dm_exec_sessions</code>, <code>sys.dm_exec_connections</code> 3. Profiler, capture and view trace events 4. View Errorlog – via TSQL but not via Management Studio 5. View built-in Management Studio Reports 6. View Activity Monitor 7. View existing SQL Server Agent jobs/schedules/steps/history 8. But, is unable to perform the following: • Backup/restore/create databases • DBCC commands • Shutdown SQL instance • Server modifications (eg. Sp_configure, create linked servers) • Database modifications (eg. Schema changes, data changes)</td>
<td></td>
</tr>
</tbody>
</table>
| 6.2  | Create read-only support staff login | Preference is to use Windows login | -- Create Windows login  
CREATE LOGIN [ReadOnlyWindowsLogin_1] FROM WINDOWS WITH DEFAULT_DATABASE = [DefaultDatabase]  
GO  
-- OR  
-- Create SQL Login with strong password  
USE [master]; CREATE LOGIN [ReadOnlySupportStaffLogin_1] WITH PASSWORD=N'`StrongPassword`', DEFAULT_DATABASE=[DefaultDatabase], DEFAULT_LANGUAGE=us_english, | |
| 6.3. | Assign permission to the read-only support staff login | Grant the login with the following permissions:  
- CONNECT SQL  
- VIEW SERVER STATE  
- VIEW ANY DATABASE  
- VIEW ANY DEFINITION  
- ALTER TRACE | Grant permissions to this user-defined server-role  
USE [master]; GRANT CONNECT SQL TO [<ReadOnlyWindowsLogin_1>];  
USE [master]; GRANT VIEW SERVER STATE TO [<ReadOnlyWindowsLogin_1>];  
USE [master]; GRANT VIEW ANY DATABASE TO [<ReadOnlyWindowsLogin_1>];  
USE [master]; GRANT VIEW ANY DEFINITION TO [<ReadOnlyWindowsLogin_1>];  
USE [master]; GRANT ALTER TRACE TO [<ReadOnlyWindowsLogin_1>];  
GO  
NOTE – requires to execute using “sysadmin” server-role member |

| 6.4. | Assign login to databases | Create and map DB user to the read-only support staff login. This includes all system databases; and subsequently all user databases. | Create user mapping to the read-only support staff login  
-- Grant view permission and view SQL errorlog via extended stored procedure  
USE [master]; CREATE USER [<ReadOnlySupportStaffUser_1>] FROM LOGIN [<ReadOnlyWindowsLogin_1>] WITH DEFAULT_SCHEMA = [<CustomSchemaName>];  
USE [master]; EXEC sp_addrolemember db_datareader, [<ReadOnlySupportStaffUser_1>];  
USE [master]; GRANT EXECUTE on xp_readerrorlog TO [<ReadOnlySupportStaffUser_1>];  
GO  
-- Grant view permission to tempdb  
USE [tempdb]; CREATE USER [<ReadOnlySupportStaffLogin_1>] FROM LOGIN [<ReadOnlySupportStaffWindowsLogin_1>] |
WITH DEFAULT_SCHEMA = [\{CustomSchemaName\}];
USE [tempdb]; EXEC sp_addrolemember
    db_datareader,
    [\{ReadOnlySupportStaffUser_1\}];
GO
-- Grant view permission to model
USE [model]; CREATE USER
    [\{ReadOnlySupportStaffUser_1\}] FROM
    LOGIN [\{ReadOnlySupportStaffLogin_1\}]
    WITH DEFAULT_SCHEMA = [\{CustomSchemaName\}];
USE [model]; EXEC sp_addrolemember
    db_datareader,
    [\{ReadOnlySupportStaffUser_1\}];
GO
-- Grant view permission to msdb
USE [msdb]; CREATE USER
    [\{ReadOnlySupportStaffUser_1\}] FROM
    LOGIN [\{ReadOnlySupportStaffLogin_1\}]
    WITH DEFAULT_SCHEMA = [\{CustomSchemaName\}];
USE [msdb]; EXEC sp_addrolemember
    db_datareader,
    [\{ReadOnlySupportStaffUser_1\}];
GO
-- Grant SQL Server Agent Reader role, but deny EXECUTE permission
USE [msdb]; EXEC sp_addrolemember
    SQLAgentReaderRole,
    [\{ReadOnlySupportStaffUser_1\}];
DENY EXECUTE ON
    [\{CustomSchemaName\}.sp_add_job to
    [\{ReadOnlySupportStaffUser_1\}];
DENY EXECUTE ON
    [\{CustomSchemaName\}.sp_add_jobschedule to
    [\{ReadOnlySupportStaffUser_1\}];
DENY EXECUTE ON
    [\{CustomSchemaName\}.sp_add_jobserver to
    [\{ReadOnlySupportStaffUser_1\}];
DENY EXECUTE ON [CustomSchemaName].sp_add_jobstep to [ReadOnlySupportStaffUser_1];
GO

-- For each user database, grant view definition to user database
USE [UserDB]; CREATE USER [ReadOnlySupportStaffUser_1] FROM LOGIN [ReadOnlySupportStaffLogin_1] WITH DEFAULT_SCHEMA = [CustomSchemaName];
GRANT VIEW DEFINITION TO [ReadOnlySupportStaffUser_1];
GO

NOTE – requires to execute using “sysadmin” server-role member
7. Secure system objects and catalog

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| 7.1. | System (and extended) stored procedures / catalogs | SQL Server 2012 has built-in extended stored procedures stored in the master database which can be called upon when requested. These built-in stored procedures are secured by default and do not require further hardening – i.e. permission should not be altered for these default database objects. All catalogs / metadata views are also secured by default. The information in the system metadata views is secured on a per-row basis. In order to be able to see system metadata for an object, a user must have some permission on the object. | The system stored procedures should not be dropped from the database; dropping these can cause problems when applying service packs. Removing the system stored procedures results in an unsupported configuration. It is usually unnecessary to completely DENY all users access to the system stored procedures, as these stored procedures have the appropriate permission checks internal to the procedure as well as external.  
- Do not remove the system stored procedures by dropping them.  
- Do not modify the default permissions on system objects.  
- Do not DENY all users/administrators access to the extended procedures.  
- Grant VIEW DEFINITION selectively at the object, schema, database, or server level to grant permission to view system metadata without conferring additional permissions. | |
## 8. Secure user objects using schemas

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Objects grouping</td>
<td>Logically grouping objects together using schemas allow for easier permission assignment</td>
<td>For each database object, ensure it is created with a non-DBO schema.</td>
<td>May require application changes</td>
</tr>
<tr>
<td>8.2</td>
<td>Object ownership and permission</td>
<td>Should manage database object security by using ownership and permission at the schema-level</td>
<td>For each database object, ensure it is created with a non-DBO schema. Once created, assign permission to schemas and not the objects directly</td>
<td>May require application changes</td>
</tr>
<tr>
<td>8.3</td>
<td>Schemas should not be owned by dbo</td>
<td>Should create separate schemas to own database objects rather than relying on the use of “dbo”, as it may contain unintended privileges</td>
<td>Not all schemas should be owned by dbo / db_owner to allow for separation of duties</td>
<td>May require application changes</td>
</tr>
<tr>
<td>8.4</td>
<td>Minimize number of owners for each schema</td>
<td>Minimize possible owner permission conflicts and ease of maintenance</td>
<td>Each schema should be owned by one owner</td>
<td>May require application changes</td>
</tr>
<tr>
<td>8.5</td>
<td>Use two-part names for database object creation and access</td>
<td>If only one-part name is used during database object creation (and access), SQL Server defaults it to dbo</td>
<td>Use two-part names for objects creation and access (eg. Schema.object).</td>
<td>May require application changes</td>
</tr>
</tbody>
</table>
9. On-going password maintenance

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1.</td>
<td>Changing service accounts</td>
<td>Changing service accounts after installation is often required to comply with various security measures. This include services such as Database engine, SQL Server Agent, Integration Services, Full-text Search, and SQL Server Browser</td>
<td>Always use SQL Server Configuration Manager to change the account used by the SQL Server or SQL Server Agent services, or to change the password for the account. In addition to changing the account name, SQL Server Configuration Manager performs additional configuration such as setting permissions in the Windows Registry so that the new account can read the SQL Server settings. Other tools such as the Windows Services Control Manager can change the account name but do not change associated settings. If the service cannot access the SQL Server portion of the registry the service may not start properly.</td>
<td>NOTE – takes effect only after SQL Server service (and Agent) is restarted</td>
</tr>
<tr>
<td>9.2.</td>
<td>Changing service accounts password</td>
<td>Depending on different security measures, service accounts password can be changed / rotated following corporate policies. This include services such as Database engine, SQL Server Agent, Integration Services, Full-text Search, and SQL Server Browser</td>
<td>Always use SQL Server Configuration Manager to change the account used by the SQL Server or SQL Server Agent services, or to change the password for the account. In addition to changing the account name, SQL Server Configuration Manager performs additional configuration such as setting permissions in the Windows Registry so that the new account can read the SQL Server settings. Other tools such as the Windows Services Control Manager can change the account name but do not change associated settings. If the service cannot access the SQL Server portion of the registry the service may not start properly.</td>
<td>NOTE – changing passwords using SQL Server Configuration Manager takes effect immediately without restarting the service</td>
</tr>
</tbody>
</table>
## 10. Data transmission encryption – database engine

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| 10.1 | Data transmission encryption | Two common methods to achieve data transmission encryption when connecting to SQL Server  
1) IPSec – for the chosen network segment  
2) SSL-based encryption  

For SSL-based encryption, SQL Server can use Secure Sockets Layer (SSL) to encrypt data that is transmitted across a network between an instance of SQL Server and a client application. The SSL encryption is performed within the protocol layer and is available to all SQL Server clients except DB Library and MDAC 2.53 clients.  

SSL can be used for server validation when a client connection requests encryption. If the instance of SQL Server is running on a computer that has been assigned a certificate from a public certification authority, identity of the computer and the instance of SQL Server is vouched for by the chain of certificates that lead to the trusted root authority. Such server validation requires that the computer on which the client application is running be configured to trust the root authority of the certificate that is used by the server. Encryption with a self-signed certificate is possible, but a self-signed certificate offers only limited protection. Self-signed certificates should be used for testing purposes only. | For #1, IPSec, because of the involvement scale, it will not be catered in this particular document.  
For #2, SSL-based encryption, the following sub-section elaborates the steps involved. | Default is disabled |
| 10.2 | Import certificate into Certificate store – Computer Account | Certificate can be from external parties; internal CA; or a self-signed certificate.  

The following lists the certificate requirements when using with SQL Server SSL encryption: | In order to place the certificate into the correct store and useable by SQL Server, it is recommended you first login to the OS using the SQL Server service account before attempting to import the certificate. |
- The certificate must be in either the local computer certificate store or the current user certificate store.
- The current system time must be after the Valid from property of the certificate and before the Valid to property of the certificate.
- The certificate must be meant for server authentication. This requires the Enhanced Key Usage property of the certificate to specify Server Authentication (1.3.6.1.5.5.7.3.1).
- The certificate must be created by using the KeySpec option of AT_KEYEXCHANGE. Usually, the certificate's key usage property (KEY_USAGE) will also include key encipherment (CERT_KEY_ENCIHERMENT _KEY_USAGE).
- The Subject property of the certificate must indicate that the common name (CN) is the same as the host name or fully qualified domain name (FQDN) of the server computer. If SQL Server is running on a failover cluster, the common name must match the host name or FQDN of the virtual server and the certificates must be provisioned on all nodes in the failover cluster.

### NOTE
Must firstly modify the service account attribute on AD to allow for “allow logon locally” before proceeding.

To import the certificate:
1. Click Start -> Run and type MMC.EXE.
2. From the File menu on the Console select Add/Remove Snap-in.
3. Select the Add button in the Add/Remove Snap-In Dialogue box.
4. Next select the Certificates Snap-in and click Add. Choose Computer Account and click Next. (SQL Server can use certificates from either stores, a user account or a computer account)
5. Select Local computer and then click Finish.
6. Click Close on the Add Snap-in Dialogue and then click OK. You should now see the certificate store for the local computer.
7. Expand to the Personal / Certificates container. This will be location for the SSL certificate.

For importing the certificate onto failover cluster SQL instances:
- Install the server certificate with the fully qualified DNS name of the failover clustered instance on all nodes in the failover cluster.
- E.G., if you have a two-node cluster, with nodes named test1.yourcompany.com and test2.yourcompany.com and a failover clustered instance of SQL Server named fcisql, you must obtain a certificate for fcisql.yourcompany.com and install the certificate on both nodes

Revert the setting on AD, and remove the SQL Service Account’s attribute of “allow logon locally”

| 10.3. | Forcing SQL Server encryption with the imported certificate | You can enable the Force Protocol Encryption option on either the server or on the client. | Enable only server-side “force protocol encryption”. | By default, both client and server side do not have force encryption. |
NOTE - Do not enable the Force Protocol Encryption option on both the client and the server. To enable Force Protocol Encryption on the server, use the SQL Server Configuration Manager. To enable Force Protocol Encryption on the client, use the SQL Server Configuration Manager on the client.

NOTE2: If you enable SSL encryption by using the SQL Native Client <version> Configuration (32 bit) or SQL Native Client <version> Configuration pages in SQL Server Configuration Manager, all connections from that client will request SSL encryption to any SQL Server to which that client connects.

If you enable Force Protocol Encryption on the server, you must install a certificate on the server.

If you want to enable Force Protocol Encryption on the client, you must have a certificate on the server and the client must have the Trusted Root Authority updated to trust the server certificate.

This allows for easier certificate management rather than deploying certificate on all connecting clients.

To enable server-side force protocol encryption:
1) Open SQL Server Configuration Manager
2) Expand SQL Server Network Configuration, on left pane, right-click on the protocols for the server instance chosen, and then click Properties
3) On the Flags tab, under General and option “Force Encryption”, change from default “No” to “Yes”
4) On the Certificate tab, configure the Database Engine to use the certificate previously imported
5) Click okay to close the dialog box.

Restart SQL Server service. Examine the SQL Server errorlog to locate the following entry, which confirms the successful usage of the SSL certificate:

<Time> <SPID> “The certificate [Cert Hash(sha1) "<Thumbprint>] was successfully loaded for encryption.

Now, start Management Studio and execute the following query, which verifies client connection is coming through using SSL:

USE master;
SELECT net_transport, auth_scheme, encrypt_option
FROM sys.dm_exec_connections (NOLOCK);
GO

Examine under the column “encrypt_option”. If it returns “TRUE” then connection to SQL Server is being encrypted. Otherwise it would return “FALSE”
### 11. Transparent database encryption – database engine

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Enable Transparent database encryption (TDE)</td>
<td>By enabling TDE, database and backup files are protected via the database encryption key, which is in-turn protected by the certificate.</td>
<td>Enabled</td>
<td>The following sub-sections elaborate the steps involved in enabling TDE. This feature is available in Enterprise-edition only. Default is not enabled.</td>
</tr>
<tr>
<td>11.2</td>
<td>Backup Service Master Key (SMK)</td>
<td>Service Master Key (SMK) is the root of all encryption components within SQL Server. It is automatically created upon SQL Server installation. Encrypt the SMK backup with a strong password.</td>
<td>-- Backup service master key USE master; BACKUP SERVICE MASTER KEY TO FILE = '&lt;Path&gt;\ServiceMasterKey.key' ENCRYPTION BY PASSWORD = '&lt;StrongPassword&gt;'; GO</td>
<td>-- Requires to execute using “sysadmin” server-role member</td>
</tr>
<tr>
<td>11.3</td>
<td>Create and backup database master key (DMK) in master database</td>
<td>Database master key (DMK) in the master database is required in order to properly utilize TDE. It is encrypted by the SMK and is used to protect the certificate. Encrypt the DMK with a strong password. Encrypt the DMK backup with a strong password.</td>
<td>-- Create master's DMK USE master; CREATE MASTER KEY ENCRYPTION BY PASSWORD = '&lt;StrongPassword&gt;'; GO</td>
<td>-- Requires to execute using “sysadmin” server-role member</td>
</tr>
<tr>
<td>11.4</td>
<td>Create or import certificate. Ensure certificates are backed up, with private key exported and protected by strong password</td>
<td>Create self-signing certificate in the MASTER database OR Create certificate based on external certificate in the MASTER database Ensure to encrypt the private key (to the certificate) with a strong password.</td>
<td>-- Create certificate USE master; CREATE CERTIFICATE &lt;SQLCert&gt; WITH SUBJECT = 'Certificate subject'; GO</td>
<td>Certificate generated by SQL Server is always 1024bit key; but TDE supports the use of 2048bit key or above. Certificate format accepted by SQL TDE is x509 DER-encoded.</td>
</tr>
</tbody>
</table>
Notice SQL Server does not examine the expiry date of a certificate on creating nor when using it. Reason being if you’re to restore a database that’s over two years old for example, and that the certificate has expired, then there is no way to restore/decrypt the database.

NOTE – This certificate is protected by the master database’s DMK.

```
'\<Path>\ExtCert.cer' WITH PRIVATE KEY (FILE = '\<Path>\ExtCert_PK.pvk', DECRYPTION BY PASSWORD = '\<StrongPassword>'),
GO
```

-- You should backup the certificate after creating it inside SQL Server
USE master;
BACKUP CERTIFICATE <SQLCert> TO FILE = '\<Path>\<SQLCert>.cer' WITH PRIVATE KEY (FILE = '\<Path>\PKtoCertWithPKPW.key', ENCRYPTION BY PASSWORD = '\<StrongPassword>'),
GO

NOTE – requires to execute using “sysadmin” server-role member

11.5. Ensure certificates have been backed up

The backup of the certificate, whether self-signing or external CA’s, once created inside SQL Server is important to ensure there is a way to decrypt the database.

-- Check certificate's last backup time
USE master;
SELECT pvt_key_last_backup_date, DB_NAME(dek.database_id) AS encrypteddatabase, c.name AS Certificate_Name FROM sys.certificates c (NOLOCK) INNER JOIN sys.dm_database_encryption_keys dek (NOLOCK) ON c.thumbprint = dek.encryptor_thumbprint
GO

NOTE – requires to execute using “sysadmin” server-role member

11.6. Separate master and user databases placement

Because certificates are used to decrypt TDE-databases, and that certificates are stored in the master database, ensure master and user databases are placed on separate drives, including backup files.

This protects against the scenario where the loss of the user database disks does not

Place master database on separate drive to all user databases. This includes the backups of master and user databases.
| 11.7. | Create database encryption key (DEK) in the user database. Use appropriate encryption algorithm encrypting with certificate | The DEK is encrypted by the server certificate and is stored on the boot page of the user database.

NOTE – this action must be performed whilst the user has db_owner privilege in <User_DB>, or with sysadmin server-role privilege | -- Create DEK
USE <User_DB>; CREATE DATABASE ENCRYPTION KEY WITH ALGORITHM = AES_192 ENCRYPTION BY SERVER CERTIFICATE <SQLCert>; GO

-- NOTE, there is no backup command to DEK.

NOTE – requires to execute using “sysadmin” server-role member |

Algorithm can be any of the following:
- AES_128
- AES_192
- AES_256
- TRIPLE_DES_3KEY |

| 11.8. | Enable TDE on the user database | The command to enable TDE is instantaneous, however, the database is not encrypted until the status of the database shows ‘3’ – meaning encryption has completed.

NOTE – this action must be performed whilst the user has db_owner privilege in <User_DB>, or with sysadmin server-role privilege | -- Enable encryption
USE master; ALTER DATABASE <User_DB> SET ENCRYPTION ON; GO

-- To check status of encryption
USE master; SELECT encryption_state FROM sys.dm_database_encryption_keys (NOLOCK) WHERE [database_id] = DB_ID ('<User_DB>'); GO

-- 2 means encryption is still in progress
-- 3 means encryption has completed |

| 11.9. | Restore TDE database to a separate server | Destination SQL Server must have the following:
1) Database master key (DMK) in the master database – does not have to be the same as the source’s
2) Create / import the exact same certificate with the same private key and password
3) Restore TDE database backup | -- Re-create master’s DMK
USE master; CREATE MASTER KEY ENCRYPTION BY PASSWORD = '<StrongPassword>'; GO

-- Re-create same certificate
USE master; CREATE CERTIFICATE <MyServerCert> FROM FILE = '<Path>\MyServerCert.cer' WITH PRIVATE KEY (FILE = |
### 11.10. Change certificate for an TDE-database

This requires the following high-level steps:

1. Create / import new certificate into SQL Server
2. Turn off encryption for existing TDE-database
3. Drop TDE-database’s database encryption key (DEK)
4. Perform backup of transaction log and database, ensure all existing log records with the old certificate have been rolled over
5. Create database encryption key (DEK) using the new certificate in the TDE-database
6. Enable encryption on the TDE-database
7. Verify the TDE-database is encrypted with the new certificate

---

```sql
-- #3, Drop the DEK which was protected by the previous certificate
USE <TDE_User_DB>
DROP DATABASE ENCRYPTION KEY;
GO
```

---

```sql
-- #4, Backup transaction log and full backup
```

---

```sql
-- #1, Create / Import certificate
USE master;
CREATE CERTIFICATE NewCert FROM FILE = '<Path>\NewCert.cer' WITH PRIVATE KEY (FILE = '<Path>\NewCert_PK.pvk', DECRYPTION BY PASSWORD = '<StrongPassword>'
GO
```

---

```sql
-- #2, Turn off encryption
USE master;
ALTER DATABASE <TDE_User_DB> SET ENCRYPTION OFF;
GO
```

---

```sql
-- #3, Drop database encryption key (DEK) using the new certificate in the TDE-database
USE <TDE_User_DB>
DROP DATABASE ENCRYPTION KEY;
GO
```

---

```sql
-- #4, Enable encryption on the TDE-database
USE <TDE_User_DB>
ALTER DATABASE SET ENCRYPTION ON;
GO
```

---

```sql
-- #5, Create database encryption key (DEK) using the new certificate in the TDE-database
USE <TDE_User_DB>
CREATE DATABASE ENCRYPTION KEY = 'NewCertKeyName'
GO
```

---

```sql
-- #6, Perform backup of transaction log and database, ensure all existing log records with the old certificate have been rolled over
USE <TDE_User_DB>
BACKUP LOG <TDE_User_DB> TO DISK = '<Path>\TDE_User_DB_log.1df'
GO
```

---

```sql
-- #7, Verify TDE-database is encrypted with the new certificate
USE <TDE_User_DB>
SELECT DATABASE_ENCRYPTION()
GO
```

---

```sql
-- #8, Create transaction log backup
USE <TDE_User_DB>
BACKUP LOG <TDE_User_DB> TO DISK = '<Path>\TDE_User_DB_log.1df'
GO
```

---

```sql
-- #9, Create full backup of TDE-database
USE <TDE_User_DB>
BACKUP DATABASE <TDE_User_DB> TO DISK = '<Path>\TDE_User_DB.bak'
GO
```

---

```sql
-- #10, Import new certificate into SQL Server
USE master;
CREATE CERTIFICATE NewCert FROM FILE = '<Path>\NewCert.cer' WITH PRIVATE KEY (FILE = '<Path>\NewCert_PK.pvk', DECRYPTION BY PASSWORD = '<StrongPassword>'
GO
```

---

```sql
-- #11, Turn on encryption for existing TDE-database
USE master;
ALTER DATABASE <TDE_User_DB> SET ENCRYPTION ON;
GO
```

---

```sql
-- #12, Verify TDE-database is encrypted with the new certificate
USE <TDE_User_DB>
SELECT DATABASE_ENCRYPTION()
GO
```

---

```sql
-- #13, Drop the DEK which was protected by the previous certificate
USE <TDE_User_DB>
DROP DATABASE ENCRYPTION KEY;
GO
```

---

```sql
-- #14, Backup transaction log and full backup
USE <TDE_User_DB>
BACKUP LOG <TDE_User_DB> TO DISK = '<Path>\TDE_User_DB_log.1df'
BACKUP DATABASE <TDE_User_DB> TO DISK = '<Path>\TDE_User_DB.bak'
GO
```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| **11.11.** To forcibly regenerate the SMK | **Only decide to do this if you suspect the keys the SMK protects has been comprised.** | **-- Regenerate SMK**

NOTE - This is resource-intensive as it re-encrypts everything underneath this SMK. | **-- OR** |
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| **11.12.** To deny disabling TDE | By default, anyone in the ‘sysadmin’ server-role will be able to enable/disable a user database’s TDE, regardless of whether the associated login/user belongs to database-role ‘db_owner’ or not. | **USE master; ALTER SERVICE MASTER KEY FORCE REGENERATE; GO**  
NOTE – requires to execute using “sysadmin” server-role member  

-- To deny the ability to disable TDE:  
-- (1) remove sysadmin server-role from login;  
-- AND  
-- (2) remove db_owner database-role from user;  
-- OR  
-- (3) if user has db_owner database-role, run the following DENY statement:  

```
USE <TDE_User_DB>; DENY ALTER TO <non_sysadmin_user> CASCADE; GO

-- To revoke the DENY, run the following:  
USE <TDE_User_DB>; REVOKE ALTER TO <non_sysadmin_user> CASCADE; GO
```

NOTE – requires to execute using “sysadmin” server-role member |
<table>
<thead>
<tr>
<th>11.13.</th>
<th>To deny backing up of certificate</th>
<th>By default, anyone in the ‘sysadmin’ server-role will be able to backup/restore certificate; OR; the login is associated to a user in the master database as ‘db_owner’</th>
</tr>
</thead>
</table>
| | | -- To deny the ability to backup/restore certificate (in the master database’s context)  
| | | -- (1) remove sysadmin server-role from login  
| | | -- AND  
| | | -- (2) remove db_owner from master database  
| | | -- OR  
| | | -- (3) if the login is associated with login in the master database with db_owner database-role, run the following DENY statement:  
| | | ```sql  
| | | USE master; DENY CONTROL to <non_sysadmin_user> CASCADE;  
| | | GO  
| | | ```  
| | | -- To revoke the DENY, run the following:  
| | | ```sql  
| | | USE master; REVOKE CONTROL TO <non_sysadmin_user> CASCADE;  
| | | GO  
| | | ```  
| | | NOTE – requires to execute using “sysadmin” server-role member  
| | | NOTE2 – by running DENY CONTROL to the <non_sysadmin_user>, that user will also be denied to perform master database backup, which contains the associated certificate as well as other server-level objects. |
12. Auditing access – database engine

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Item / Parameter</th>
<th>Description</th>
<th>Recommended settings</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1</td>
<td>Login auditing</td>
<td>Auditing for account login must be enabled</td>
<td>Enabled for all login attempts</td>
<td>Default is to audit “failure” logins only</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Can be changed in Management Studio:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Right-click on server instance, choose the “Facet” option</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Under the Facet dropdown, choose “Server Audit”</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Examine the value of “LoginAuditLevel”</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Set value from “Failure” to “All”</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOTE – requires to execute using “sysadmin” server-role member</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NOTE2 – SQL service must be restart to become effective</td>
<td></td>
</tr>
<tr>
<td>12.2</td>
<td>Server-level auditing</td>
<td>Audit events that occur on server-level.</td>
<td>Both server audit (destination) and server audit specification (events) must be specified beforehand.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1) To create server audit, execute the following in Management Studio:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>USE master; CREATE SERVER AUDIT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>&lt;Server_Audit&gt;</code> <strong>TO APPLICATION_LOG</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>WITH ( QUEUE_DELAY = 1000, ON_FAILURE = SHUTDOWN );</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>GO</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2) To create server audit specification, execute the following in Management Studio:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>USE master; CREATE SERVER AUDIT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>SPECIFICATION <code>&lt;Server_Audit_Spec&gt;</code></strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>FOR SERVER AUDIT <code>&lt;Server_Audit&gt;</code></strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>ADD (AUDIT_CHANGE_GROUP),</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>ADD (DBCC_GROUP),</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>ADD (FAILED_LOGIN_GROUP),</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>ADD (SERVER_PRINCIPAL_CHANGE_GROUP),</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This is available on Enterprise-edition only</td>
<td></td>
</tr>
</tbody>
</table>
ADD (SERVER_ROLE_MEMBER_CHANGE_GROUP),
ADD (SUCCESSFUL_LOGIN_GROUP),
ADD (TRACE_CHANGE_GROUP)
WITH (STATE=ON)
GO

USE master; ALTER SERVER AUDIT <Server_Audit> WITH (STATE = ON);
GO

NOTE – requires to execute using “sysadmin” server-role member

NOTE2 – additional auditing specifications can be added but be aware of performance overheads

NOTE3 – any changes to auditing or specifications required restart of the audit

### 12.3. Database-level auditing

Audit events that occur on database-level

Similar to server-level audit, there are two parts of database auditing that must be completed prior.

1) Create a new server audit (destination), which is the same step as the above section “server-level auditing”; or specifying an existing server audit destination

2) Create a new database audit specification, by executing the following in Management Studio:

```sql
USE <DB_NAME>; CREATE DATABASE AUDIT SPECIFICATION <DB_AUDIT_SPEC> FOR SERVER AUDIT <Server_Audit>
ADD (SCHEMA_OBJECT_CHANGE_GROUP),
ADD (DATABASE_PERMISSION_CHANGE_GROUP),
ADD (SCHEMA_OBJECT_PERMISSION_CHANGE_GROUP),
ADD (DATABASE_OBJECT_PERMISSION_CHANGE_GROUP),
ADD (DATABASE_PRINCIPAL_CHANGE_GROUP)
WITH (STATE = ON);
GO
```

This is available on Enterprise-edition only
| NOTE – requires to execute using “sysadmin” server-role member |
| NOTE2 – additional auditing specifications can be added but be aware of performance overheads |
| NOTE3 – any changes to auditing or specifications required restart of the audit |