STIG Monitor Installation

Contents

[2 Prerequisites 3](#_Toc465253136)

[2.1 Checklist of Items Required 3](#_Toc465253137)

[2.2 Auditing Storage Requirements 4](#_Toc465253138)

[3 Installation Summary 5](#_Toc465253139)

[3.1 SQL Audit Section 5](#_Toc465253140)

[3.1.1 Audit setup on the Main Server 5](#_Toc465253141)

[3.1.2 Audit Setup on each Remote Monitored Server 5](#_Toc465253142)

[3.2 Policy Section 6](#_Toc465253143)

[4 Installation Scripts 6](#_Toc465253144)

[4.1 Files 6](#_Toc465253145)

[4.2 Running the Scripts 7](#_Toc465253146)

[5 Installation Steps 13](#_Toc465253147)

[5.1 Audit Setup on the Central Management Server 13](#_Toc465253148)

[5.1.1 Create the audit database, tables, and stored procedures. Link 13](#_Toc465253149)

[5.1.2 Create the folders and the share to hold the audit files on the Main Server. Link 14](#_Toc465253150)

[5.1.3 Create the SQL instance Audit, the instance Audit specification, and the Database Audit. specifications. Link 18](#_Toc465253151)

[5.1.4 Create the Proxy and Credential. (02CreateAuditObjects.sql) Link 19](#_Toc465253152)

[5.1.5 Create the Import Job which will move the audit data from folders on the Main Server into the repository database. (03CreateAuditImportFilesjob.sql) Link 20](#_Toc465253153)

[5.1.6 Create the AuditCopy job that copies the audit data from the folder where the files are created to the processing folder. (04CreateAuditCopyJob.sql) Link 21](#_Toc465253154)

[5.1.7 Publish the reports to the ReportServer. Link 22](#_Toc465253155)

[5.1.8 Set the permissions to view Audit information through the included reports. Link 25](#_Toc465253156)

[5.2 Audit Setup on each Monitored Server Instance 25](#_Toc465253157)

[5.2.1 Create the SQL instance Audit, the instance Audit specification, and the Database Audit specification. (02CreateAuditObjects.sql) Link 25](#_Toc465253158)

[5.2.2 Create the Proxy and Credential (02CreateAuditObjects.sql) Link 26](#_Toc465253159)

[5.2.3 Create the AuditCopy job that copies the audit data from the folder where the files are created to the correct folder on the Main Server. (04CreateAuditCopyJob.sql) Link 27](#_Toc465253160)

[5.3 Policy Management Setup 28](#_Toc465253161)

[5.3.1 Manually Import Policies (SQL2012v10 STIG Policies.zip) Link 28](#_Toc465253162)

[5.3.2 Create the Database Objects and insert data (06CreatePolicyDB.sql) Link 31](#_Toc465253163)

[5.3.3 Configure the PowerShell user account rights Link 31](#_Toc465253164)

[5.3.4 Create Synonyms, Linked Server, Login, Credential, and Proxy (each monitored server) (07CreatePolicyObjects.sql) Link 34](#_Toc465253165)

[5.3.5 Configure an Alias for the CMS Instance Link 35](#_Toc465253166)

[5.3.6 Manual policies test. Link 37](#_Toc465253167)

[5.3.7 Disable the auto-created evaluation job Link 40](#_Toc465253168)

[5.3.8 Create the working directories Link 41](#_Toc465253169)

[5.3.9 Create the Import Policies Job Link 43](#_Toc465253170)

[5.3.10 Publish reports, data source, and shared dataset to SSRS (Report Server only) (09PublishPolicyReports.rss) Link 45](#_Toc465253171)

[6 Adding a new Server Group to the System for Evaluation 46](#_Toc465253172)

[6.1.1 Create a new folder under X:\SQLPolicy for the new Server Group 46](#_Toc465253173)

[6.1.2 Either add a step to the Import Policy Job or create a new Import Policy job 46](#_Toc465253174)

[7 Adding Instances to an Existing Monitored CMS Group 48](#_Toc465253175)

[7.1.1 Go through the Checklist of Items Required for Monitored Instances. 48](#_Toc465253176)

[7.1.2 Create the folders for the new instance’s audit files 48](#_Toc465253177)

[7.1.3 Create the SQL instance Audit, the instance Audit specification, and the Database Audit specification. (02CreateAuditObjects.sql) Link 49](#_Toc465253178)

[7.1.4 Create the Proxy and Credential (02CreateAuditObjects.sql) Link 50](#_Toc465253179)

[7.1.5 Create the AuditCopy job that copies the audit data from the folder where the files are created to the correct folder on the Main Server. (04CreateAuditCopyJob.sql) Link 50](#_Toc465253180)

[7.1.6 Edit the “Import Audit Files” job in the SQL Agent of the Main Server to include the new server. 51](#_Toc465253181)

[7.1.7 Create Synonyms, Linked Server, Login, Credential, and PolicyProxy (07CreatePolicyObjects.sql) 53](#_Toc465253182)

[8 Appendix A – Reports 54](#_Toc465253183)

[9 Appendix B – Auditing for STIGs Detail 56](#_Toc465253184)

[1.1 Required Trace Items 59](#_Toc465253185)

[1.2 Trace Event ID mapping SQL Audit Action Group 61](#_Toc465253186)

# Prerequisites

## Checklist of Items Required

* SQL 2012 Management Studio with access to every SQL Instance to be audited.
* Pre-installed instance of SQL 2012 Management Studio with access to every SQL instance to be audited.

Server Name where installed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Gather a list of SQL Instances to monitor. Group them by major version number (2008, 2008 R2, 2012, 2014). If there are no plans to use the optional CMS Server Groups, the rest of this bullet item can be skipped.

An early version of this demonstration pulled instance names from Central Management Server (CMS) groups. That functionally is available with a small amount of additional configuration.

The CMS feature requires the main server rto be running SQL 2012 or 2014 Enterprise Edition. To use this feature, configure the main server as a Central Management Server. Create Groups named after the SQL Versions (2008, 2008 R2, 2012, 2014) – The CMS server requires SQL Server 2012 Enterprise Edition or later. Adding the servers to the groups will be covered in these instructions.

CMS Server Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CMS Server IP Address and Port #:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

An Alias Name for the CMS Server Instance:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* All SQL Servers have Kerberos authentication enabled. This includes having proper SPNs registered with Active Directory for its domain. For more information, see Registering a Service Principal Name (http://msdn.microsoft.com/en-us/library/cc280459.aspx) and Service Principal Name (SPN) Client Connections (http://msdn.microsoft.com/en-us/library/ms191153.aspx).
* SQL Agent enabled and running on the Main Instance using a domain or service account specific to that SQL Agent.

Main Server’s SQL Agent Account: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Name and password of Domain Account used to access All SQL Servers.

SQL Admin Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Note: Don’t write down any passwords!

* Name and password of a domain user account for use in running SQL Agent Audit jobs. This account will need permissions to copy files from servers back to the Main Server.

SQL Audit Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Name and password of a domain user account for use in reading audit data from the SQL server database for SQL Audit Reports. This account will need permissions on the Main Server’s Reporting services only.

SQL Audit Report Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Name and password of a domain user account for use in running the PowerShell script for policy evaluation. Can be the same as the audit account above. [Link](#_Configure_the_PowerShell)

SQL PowerShell Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Local or SAN attached drive on each server to store the audit files for each instance. Create a list of these audit file locations by SQL instance (Example and table below). 5 GB per monitored server is a suggested minimum for moderately busy servers.
* Local or SAN attached storage on the Main Server to store audit files that are copied to the Main Server from each monitored server. 5 GB per monitored instance is a suggested minimum for moderately busy servers. Naming this folder “AuditSQLAudit” is highly recommended.

Drive and Storage Path: \_\_\_:\SQLAudit\\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Auditing Storage Requirements

When Auditing is enabled, the SQL Instance listens for the requested auditable events, and when those events fire, a record is logged to a file. The target location for a SQL Audit file can be physical or UNC. If the SQL Instance is a stand-alone instance, it doesn’t really matter what file path is chosen for the Audit Log file, so long as there is sufficient space. If the SQL Instance is a Failover Cluster the best choice may be a file share.

The DBA has control over the local audit files that each instance will generate. As described in the next section, each audit file can be limited in size, can be limited in the total number of files that the instance will administer, and finally, if and when the files should roll-over. Setting up SQL Audit with this Build Guide, the only storage requirement needed is the location that the audit files will be written to.

* If the server is a stand-alone server with few users, there will be light storage requirements.
* The enterprise SharePoint portal SQL Servers, will likely have a significant storage requirement for the SQL Audit files.
* As a starting point for auditing, consider provisioning:
  + Low usage SQL Instances 10 GB per Instance for the audit files.
  + Medium-Low usage SQL Instances 20 GB per Instance for the audit files.
  + Medium usage SQL Instances 35 GB per Instance for the audit files.
  + Medium-High usage SQL Instances 60 GB per Instance for the audit files.
  + High usage SQL Instances 200 GB per Instance for the audit files.

# Installation Summary

## SQL Audit Section

### Audit setup on the Main Server

1. Create the audit database, tables, and stored procedures. [Link](#_Create_the_audit)
2. Create the folders and the share to hold the audit files on the Main Server. [Link](#_Create_the_folders)
   1. Create x:\SQLAudit
   2. Create one sub folder for each instance on the Main Server to be monitored like x:\SQLAudit\Server\_InstanceName (named instances) or x:\SQLAudit\_MSSQL (default instances). This is not for servers other than the Main Server. We’ll get to that later.
   3. Assign Privileges.
3. Create the SQL instance Audit, the instance Audit specification, and the Database Audit specifications by running 02CreateAuditObjects.sql. [Link](#_Create_the_SQL_1)
4. Create the Import Job which will move the audit data from folders on the Main Server into the repository database by running 03CreateAuditImportFilesjob.sql. [Link](#_Create_the_Import_1)
5. Create the AuditCopy job that copies the audit data from the folder where the files are created to the Main Server’s holding folder created in 2b above by running 04CreateAuditCopyJob.sql. [Link](#_Create_the_AuditCopy_1)
6. Publish the reports to the ReportServer. [Link](#_Publish_the_reports)
   1. This is a script to be used with the rs.exe cmd line utility.
   2. See in-script comments to set values at the top of the .rss script.
7. Set the permissions to view Audit information through the included reports. [Link](#_Set_the_permissions)
   1. Use sysadmin and administrator accounts for now. Will update this shortly.

### Audit Setup on each Remote Monitored Server

1. Create the SQL instance Audit, the instance Audit specification, and the Database Audit specification. (02CreateAuditObjects.sql) [Link](#_Create_the_SQL)
2. Create the Proxy and Credential (02CreateAuditObjects.sql) [Link](#_Create_the_Proxy)
3. Create the AuditCopy job that copies the audit data from the folder where the files are created to the correct folder on the Main Server. This job is created and run on each remote server (04CreateAuditCopyJob.sql) [Link](#_Create_the_AuditCopy)
4. (Optional) Edit the “Import Audit Files” job to include the new server if necessary. [Link](https://microsoft-my.sharepoint.com/personal/curtisk_microsoft_com/Documents/SQL%20STIGs/AnneLabbe/Import#_Edit_the_)

## Policy Section

All steps are performed on the Main Server only unless otherwise marked.

1. Manually Import Policies (SQL2012v5 STIG Policies) [Link](#_Manually_Import_Policies)
2. Configure the PowerShell user account rights (give it Admin for now) [Link](#_Configure_the_PowerShell)
3. Create the Database (06CreatePolicyDB.sql) [Link](#_Create_the_Database)
4. Create Synonyms, Linked Server, Login, Credential, and Proxy (each monitored server) (7) (07CreatePolicyObjects.sql) [Link](#_Create_Synonyms,_Linked)
5. Create an Alias for the Main Server [Link](#_Configure_Alias_for)
6. Manual policies test. Evaluate through SSMS. Make sure targets are hit. [Link](#_Manual_policies_test.)
7. Disable the auto-created evaluation job [Link](#_Disable_the_auto-created)
8. Create the working directory [Link](#_Create_the_working)
   1. x:\SQLPolicy\Working should work fine
9. Create the Import Policies Job (08CreatePolicyImportJob.sql) [Link](#_Create_the_Import)
10. Publish reports, data source, and share dataset to SSRS (Report Server only) (09PublishPolicyReports.rss) [Link](#_Publish_reports,_data)
    1. This is a script to be used with the rs.exe cmd line utility.
    2. See comments at the top of the .rss script.

# Installation Scripts

## Files

The following files are needed for the installation of the SQL Monitor:

|  |
| --- |
| File Names |
| 01CreateAuditDB.sql |
| 02CreateAuditObjects.sql |
| 03CreateAuditImportFilesjob.sql |
| 04CreateAuditCopyJob.sql |
| 05PublishAuditReports.rss |
| 06CreatePolicyDB.sql |
| 07CreatePolicyObjects.sql |
| 08CreatePolicyImportJob.sql |
| 09PublishPolicyReports.rss |
| ImportPolicies.ps1 |
| EvalPolicies.ps1 |
| SQL2012v10 STIG Policies.zip |
| STIGAuditReports.zip |
| STIGPolicyReports.zip |

The eleven files that start with consecutive numbers are used in that order to install the system. Eight are SQL scripts, two are RSS files for installing reports and one is a PowerShell scripts. Their use and variable descriptions is described below.

Additionally, there are 32 files for the Audit reports, 18 for the Policy reports, and 178 policy definition files that will be explained in their relative configuration sections below.

## Running the Scripts

The scripts provided for the STIG Monitor are intended to be run from the SQL Server Management Studio (SSMS) from query windows in SQLCMD mode.

**To open a script in SSMS**: Select the File menu and then Open and finally File… Navigate to the desired .sql file, select it and click the Open button. The contents of the file will open in a new query window.

**To put the query window into SQLCMD mode**: With the query window open and the focus on that window, select the Query menu and choose SQLCMD mode which will be close to the bottom of the long dropdown menu. Once you’ve selected this option, the variables in the script will be shown against a gray background. This is a sign you are in the correct mode.

:setvar AuditDBName "SQLAudit"

:setvar AuditDBDataFilePath "C:\{Path}\DATA"

**To set the proper variable values**: All of the scripts will have variables at the top of the script. Some of these will need to be set for your environment or to include the server names, instance names, or other desired values. Note in the screen clip above, the names of the variables are AuditDBName and AuditDBDataFilePath and the values for those variables are “SQLAudit” and “C:\{Path}\DATA ”. The values are already in their proper format (quoted). Do not change the quotes. Some of the values will be drive locations and paths to folders. The example values will not end with a backslash (“\”) and the value that you enter should not either. Also, some values will appear in more than one script. That value must stay the same between scripts.

Use the table below to note the values you plan to use during the installation. Variables that appear in multiple scripts will be denoted by an asterisk.

|  |  |  |
| --- | --- | --- |
| Variable | Script(s) | Description |
| AuditDBName: | 01CreateAuditDB.sql  04CreateAuditImportJob.sql | The name of the Audit database. Highly recommend using "SQLAudit" |
|  | | |
| AuditDBDataFilePath: | 01CreateAuditDB.sql | The Location where the data files will be created. Should not end with "\" |
|  | | |
| AuditDBLogFilePath: | 01CreateAuditDB.sql | The Location where the log files will be created. Should not end with "\" |
|  | | |
| AuditRetentionTimeLow: | 01CreateAuditDB.sql | Number of DAYS to keep data from Audit\_Login and Audit\_Logout. Recommend starting with the default of 2 |
|  | | |
| AuditRetentionTimeMed: | 01CreateAuditDB.sql | Number of WEEKS to keep data from Audit\_backup\_restore, Audit\_DBCC, Audit\_database\_change, Audit\_trace\_change, Audit\_object\_change, Audit\_server\_operation, and Audit\_database\_principal\_change. Recommend starting with the default of 1 |
|  | | |
| AuditRetentionTimeHigh: | 01CreateAuditDB.sql | Number of MONTHS to keep Audit\_failed\_login audit data. Recommend starting with the default of 1 |
|  | | |
| AuditRetentionTimeYear: | 01CreateAuditDB.sql | Audit\_schema\_object\_permission\_change, Audit\_server\_principal\_change, Audit\_server\_permission\_change, Audit\_login\_change\_password, Audit\_server\_role\_member\_change, Audit\_database\_role\_member\_change, Audit\_application\_role\_password\_change, Audit\_schema\_object\_change, Audit\_application\_role\_password\_change, Audit\_schema\_object\_change, Audit\_application\_role\_password\_change, Audit\_schema\_object\_change, Audit\_server\_principal\_impersonation, Audit\_object\_ownership\_change, and Audit\_change data. Recommend starting with the default of 1. |
|  | | |
| AuditFilePath: | 02CreateAuditObjects.sql | The target location for SQL Audit Log File: Physical or UNC. |
|  | | |
| SourceInstance: | 02CreateAuditObjects.sql 04CreateAuditCopyJob.sql | The name of the monitored instance. "" or "MSSQL" for default instances |
|  | | |
| IsMirrored: | 02CreateAuditObjects.sql | Does the target SQL Server instance participate in a Mirror Set [ "True" | "False" ]]. IMPORTANT: The GUIDs in this script can only be used to uniquely identify one Mirror Set in an enterprise. New GUIDs must then be assigned to each Audit before this script can be used for each subsequent Mirror Set. |
|  | | |
| MaxSize: | 02CreateAuditObjects.sql | The maximum log file size (in MB's) before rollover: (integer) |
|  | | |
| MaxRolloverFiles: | 02CreateAuditObjects.sql | The maximum number of rollover files: (integer) |
|  | | |
| ReserveDiskSpace: | 02CreateAuditObjects.sql | Should disk space be reserved for log files [ "ON" | "OFF" ]. |
|  | | |
| QueueDelay: | 02CreateAuditObjects.sql | The amount of time (in Milliseconds) with which events are written to the log file asynchronously; a value of (0) zero causes events to be written in sychronous mode. |
|  | | |
| OnFailure: | 02CreateAuditObjects.sql | Determines if the server should be shut down if audit events cannot be written to the log file(s): [ SHUTDOWN | CONTINUE ]. |
|  | | |
| EnableState: | 02CreateAuditObjects.sql | Determines if the Audits and Audit Specifications should be enabled or disabled after creation: [ ON | OFF ]. |
|  | | |
| AuditCredAcct: | 02CreateAuditObjects.sql | The login with domain name to be used for SQL Agent Audit jobs |
|  | | |
| AuditCredPW: | 02CreateAuditObjects.sql | The password for the above account. Recommended to put in a holding password (not the real password) and change the credential’s password after running the script. Better security than putting passwords in plain text. |
|  | | |
| AuditCredObj: | 02CreateAuditObjects.sql | The name of the credential object. Recommend not changing this value. |
|  | | |
| AuditProxy: | 02CreateAuditObjects.sql  04CreateAuditImportJob.sql  04CreateAuditCopyJob.sql | The name of the SQL Agent proxy. Recommend not changing this value. |
|  | | |
| MaxFiles | 02CreateAuditObjects.sql | The max number of files processed per job run. Only used with PowerShell file copy method (see comments in script) |
|  | | |
| RenamedSA: | 03CreateAuditImportFilesjob.sql  04CreateAuditCopyJob.sql | The current name of the "sa" account. STIGs require it to be renamed and disabled. That is the preferred state for this use. |
|  | | |
| ServerList: | 03CreateAuditImportFilesjob.sql | This is a list of the initial servers being audited separated by commas. One copy job step will be created for each server or server\instance pair in the list. |
|  |  |  |
| WorkingPath: | 03CreateAuditImportFilesjob.sql | The working path for processing audit output files. Recommend using "Drive:\SQLAudit" |
|  | | |
| WorkingFolder: | 03CreateAuditImportFilesjob.sql | The folder of the working folder, located in the WorkingPath above. |
|  | | |
| MaxFilesProcessed | 03CreateAuditImportFilesjob.sql | The max number of files processed per job run. Only used with PowerShell file copy method (see comments in script) |
|  | | |
| SourceServer: | 04CreateAuditCopyJob.sql | The name of the monitored server |
|  | | |
| SourcePath: | 04CreateAuditCopyJob.sql | Location where the audit files are created and initially stored. |
|  | | |
| MainServer: | 04CreateAuditCopyJob.sql  07CreatePolicyObjects.sql  08CreatePolicyImportJob.sql | The name of the Main Server. Do not include the instance name. |
|  | | |
| DestShare: | 04CreateAuditCopyJob.sql | The share on the Main Server that is the target of the copy. Strongly recommend using "SQLAudit" |
|  | | |
| PolicyDBName: | 06CreatePolicyDB.sql  10CreatePolicyImportJob | Name given to the Policy Database. Highly Recommend "MDW" |
|  | | |
| PolicyDBDataFilePath: | 06CreatePolicyDB.sql | The Location where the data files will be created. Should not end with "\" |
|  | | |
| PolicyDBLogFilePath: | 06CreatePolicyDB.sql | The Location where the log files will be created. Should not end with "\" |
|  | | |
| ServerList: | 06CreatePolicyDB.sql | This is a list of the remote server instances being evaluated, separated by commas. It different from the list 03CreateAuditImportFilesJob in that the Main Server Instance should not be included in this list. |
|  |  |  |
| MainInstName: | 07CreatePolicyObjects.sql  08CreatePolicyImportJob.sql | The name of the Instance of the Main Server |
|  | | |
| PolicyCredAcct: | 07CreatePolicyObjects.sql | The login to be used in running PowerShell scripts |
|  | | |
| PolicyCredPW: | 07CreatePolicyObjects.sql | The PowerShell login's password. Recommend entering a holding password and changing to the correct password using the GUI after running this script. |
|  | | |
| PolicyCredObj: | 07CreatePolicyObjects.sql | The name given to the credential. No need to change. |
|  | | |
| PolicyProxy: | 07CreatePolicyObjects.sql | The name given to the SQL Agent Proxy. No need to change. |
|  | | |
| OwnerLogin: | 08CreatePolicyImportJob.sql | The login that owns the SQL Agent job. The "sa" account is recommended, even if disabled. Use the new name for sa if it has been renamed. |
|  | | |
| EvalScriptPath: | 08CreatePolicyImportJob.sql | The Location where the EvalScriptFile is stored. Should not end with "\" |
|  | | |
| EvalGroupName: | 08CreatePolicyImportJob.sql | The name of the Instance Group to be evaluated. |
|  | | |
| EvalScriptFile: | 08CreatePolicyImportJob.sql | The name of the PowerShell script that evaluates the policies |
|  | | |
| JobCategory | 08CreatePolicyImportJob.sql | Category of the SQL Agent job. No need to change. |
|  |  |  |
| GroupSqlVer | 08CreatePolicyImportJob.sql | Version of SQL Server in the group being evaluated. 2012 and 2014 are valid values. |

**How to use the RSS files**: There are two .RSS files included in the setup scripts. They are 05PublishAuditReports.rss and 09PublishPolicyReports.rss. They are used with the rs.exe utility to deploy reports directly to the ReportServer service. They are run from an administrative command prompt. Here is an example command line:

rs -i 05PublishAuditReports.rss -s <http://RptServerName/ReportServer>

Each RSS file has several variables at the top of the script that need to be set for your environment prior to running. They look like the following in the script:

Dim SvrInst as String = "DBServer\Instance"

Dim PrjDB As String = "SQLAudit"

Those that need to possibly be changed are:

|  |  |  |
| --- | --- | --- |
| Variable | Value Examples | Description |
| SvrInst | "DBServer\Instance" | Main Server and Instance Name |
|  |  |  |
| PrjDB | "SQLAudit" | Same as AuditDBName variable above |
|  |  |  |
| rptrptPath | "C:\LocationOfRDLFiles" | Location of .RDL files |

**Use of PowerShell scripts**: The provided PowerShell scripts, EvalPolicies.ps1 and ImportPolicies.ps1, are used within SQL Agent jobs. Their configuration into the system is described in the relevant sections below.

# Installation Steps

## Audit Setup on the Central Management Server

### Create the audit database, tables, and stored procedures. [Link](#_On_the_Central)

1. Open the 01CreateAuditDB.sql script in an SSMS query window with a connection to the Main Server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the files location for the data and log file of the database. The variables and default values from the script are:

:setvar AuditDBName "SQLAudit"

:setvar AuditDBDataFilePath "C:\{Path}\DATA"

:setvar AuditDBLogFilePath "C:\ {Path}\DATA"

:setvar AuditRetentionTimeLow 2

:setvar AuditRetentionTimeMed 2

:setvar AuditRetentionTimeHigh 1

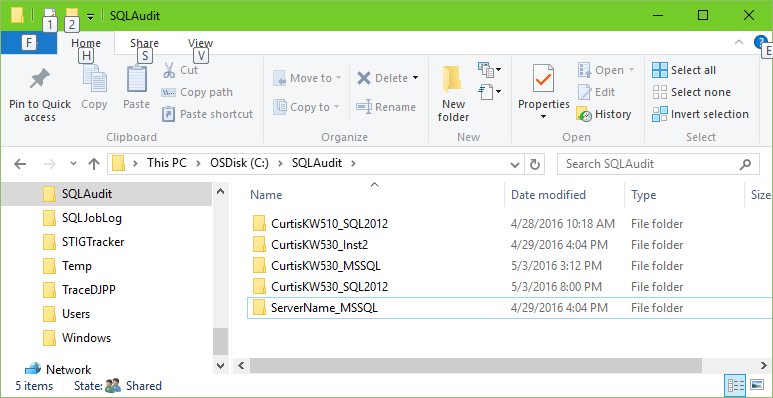
:setvar AuditRetentionTimeYear 1

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the ‘SQLAudit’ database.
   * In the SSMS Object Explorer, refresh the databases folder.
   * Expand the list of databases.
   * Confirm the existence of the new ‘SQLAudit’ database.

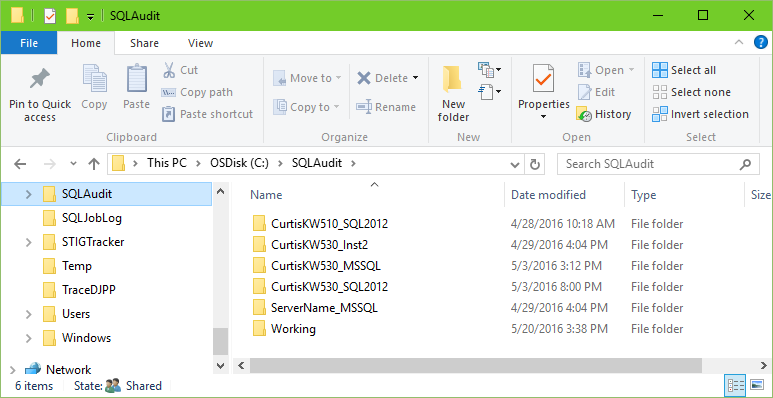
### Create the folders and the share to hold the audit files on the Main Server. [Link](#_On_the_Central)

NOTE: Files are copied to these folders from each of the server’s folders.

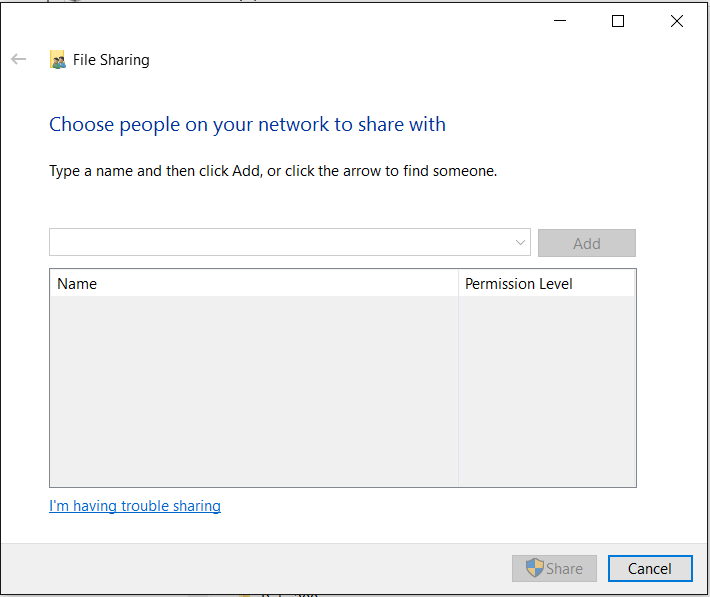
1. On a drive with over 20 GB of free space, create a folder called ‘SQLAudit’
   * Note: You will need this name in some of the upcoming scripts. Keep it handy.
2. Under this folder create a new folder for each SQL instance to be monitored as shown in the image below.
   * Name each subfolder after the Server Name and Instance name with an underscore between the two. Example: MyServer\_SQL2012.
   * Important: For default instances, use MSSQL for the instance name. Example: MyServer\_MSSQL



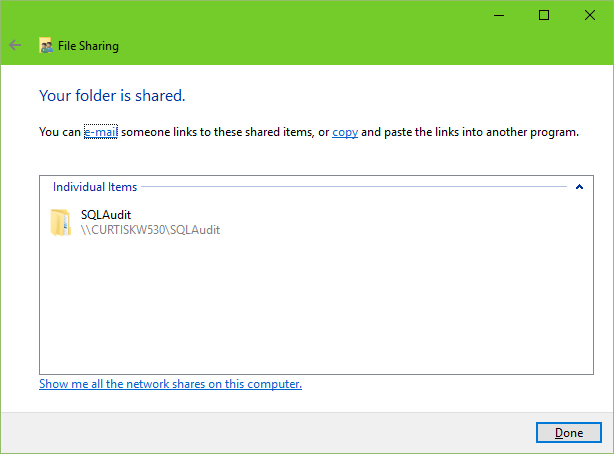
1. In File Explorer under the SQLAudit folder, create a folder called ‘**Working’** for the Audit Import job to use.



1. Create a share for the ‘SQLAudit’ directory created above.
   * In Windows Explorer, right-click on the ‘SQLAudit’ folder and choose ‘Share With…’. The following dialog box will appear.



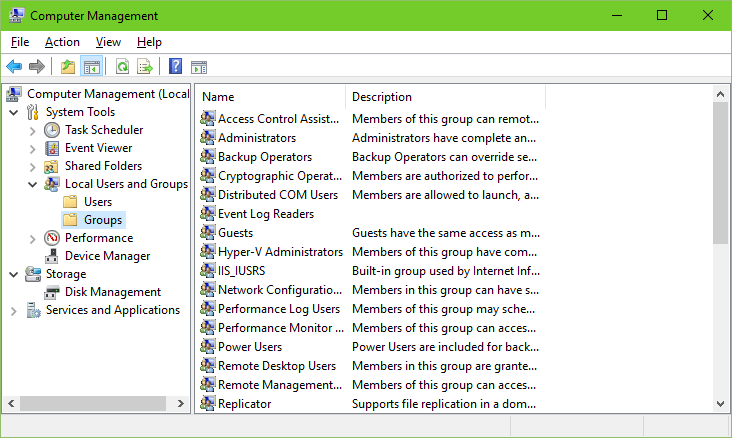
* Add Read/Write permissions for the Main Server’s SQL Agent Service account.
* Click “Share”to accept.



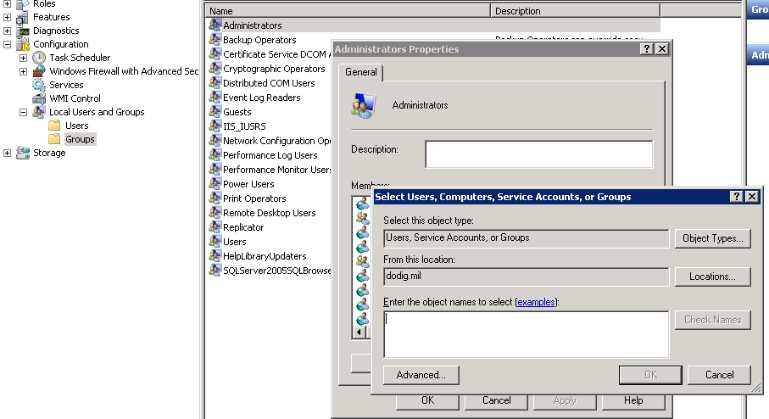
* Click “Done” to complete.

1. Add Permissions for the Audit Proxy Account

* In the Windows Explorer window, right-click on the node for the local computer (“My Computer” by default) and choose “Manage”
* Open the “Server Manager” (or Computer Management) node, then the “Configuration” (could be System Tools)) node, then the “Local Users and Groups” node, and finally, click on the “Groups” node.



* Double click on the “Administrators” group name to open the “Administrators Properties” box.
* Click the “Add” button.



* Type in the name of the audit proxy account. (See [Prerequisites](#_Prerequisites) above)
* Click the “Check Names” button to verify.
* Click “OK” to accept.

### Create the SQL instance Audit, the instance Audit specification, Database Audit. specifications, Audit Proxy, and Audit Credentials. [Link](#_On_the_Central)

1. Open the 02CreateAuditObjects.sql script in an SSMS query window with a connection to the Main Server Instance.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the files location for the files of the audit (AuditFilePath) and the Audit Credential Account (AuditCredAcct). Note that the **AuditFilePath** is a local holding place for the local audit output. The **SQLAudit** folder created above will only exist on the Main Server and will be used in processing all audit data into the database for reporting. They are two different folders. The variables and default values from the script are:

:setvar AuditCredAcct "DOMAIN\UserName"

:setvar AuditCredPW "PasswordHere"

-- Recommend using wrong PW here and changing in UI (more secure)

-- Probably no need to change the values below

:setvar AuditFilePath "C:\Audit"

:setvar InstanceName "InstanceName"

:setvar IsMirrored "False"

:setvar MaxSize 100

:setvar MaxRolloverFiles 255

:setvar ReserveDiskSpace "OFF"

:setvar QueueDelay 1000

:setvar OnFailure CONTINUE

:setvar EnabledState ON

:setvar AuditCredObj "AuditCredential"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the instance audit, instance audit specification, and the database audit. specifications.
   * In the SSMS Object Explorer, refresh the Security node.
   * In the Security node, expand the Audits node and confirm the existence of the audit named STIG\_COMPLIANT\_SERVER\_AUDIT.
   * In the Server Audit Specifications node, expand the node and confirm the existence of the audit specification named STIG\_COMPLIANT\_SERVER\_AUDIT
   * (Enterprise Edition only) In any of the user database, expand the database node, the Security node, and the Database Audit Specification node to confirm the existence of an audit specification named STIG\_COMPLIANT\_DATABASE\_AUDIT
   * In the Windows explorer, check for the existence of the AuditFilePath as specified in the AuditFilePath script variable.
   * In some versions of SQL Server covered by this utility, database audits are not available. These are not required for the STIGs.
   * In the SSMS Object Explorer, refresh the Security node and expand the Credentials node to confirm the existence of the “AuditCredential”. If you entered an incorrect password for the credential in the creation script, open the credential’s properties dialog and change the password.
   * In the SSMS Object Explorer, refresh the SQL Agent node and expand the SQL Agent and Proxies nodes for Operating System (CmdExec) and PowerShell to confirm the existence of the “AuditProxy”.

### Create the Import Job which will move the audit data from folders on the Main Server into the repository database. (03CreateAuditImportFilesjob.sql) [Link](#_On_the_Central)

Note: You will need to create one INSERT statement for each monitored instance in Step #3.

1. Open the 03CreateAuditImportFilesjob.sql script in an SSMS query window with a connection to the Main Server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the RenamedSA, ServerList and WorkingPath values. The variables and default values from the script are:

:setvar RenamedSA "nosa"

:setvar ServerList "Server\Inst, Server2,Server2\Inst"

:setvar WorkingPath "C:\SQLAudit"

:setvar WorkingFolder "Working"

:setvar AuditDBName "SQLAudit"

:setvar AuditProxy "AuditProxy"

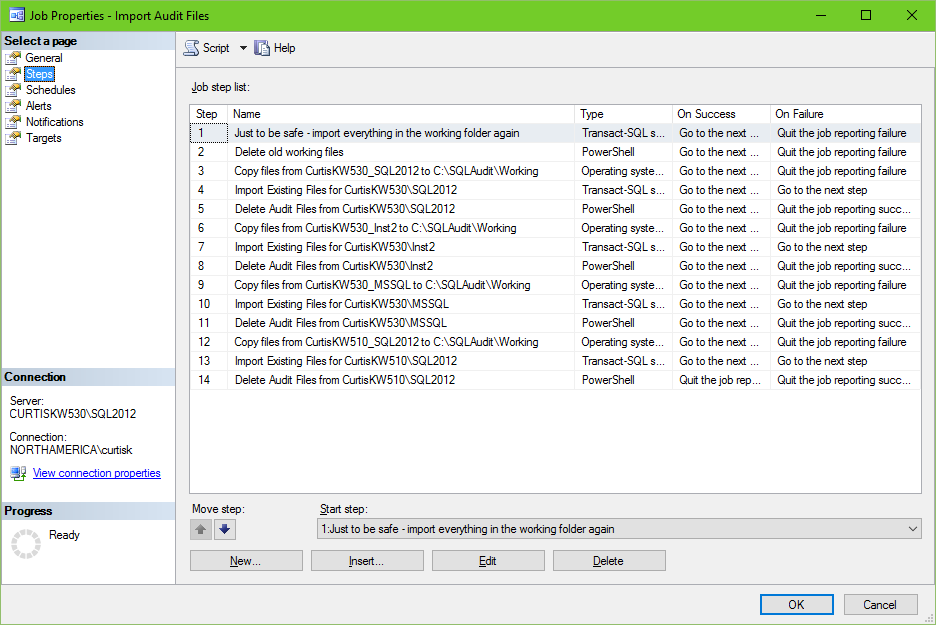
:setvar MaxFilesProcessed "5"

-- The 2 variables below are only used if the archive steps below are uncommented. This option is not fully functional in this version.

:setvar FullArchivePath "\\ArchiveName\Share"

:setvar ArchiveRetentionDays "30"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of a SQL Agent job named “Import Audit Files”
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “Import Audit Files” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that three steps (Copy Files, Import Existing Files, Delete Audit Files) exist for each of the instances to be monitored. It should look something like the screenshot below.



### Create the AuditCopy job that copies the audit data from the folder where the files are created to the processing folder. (04CreateAuditCopyJob.sql) [Link](#_On_the_Central)

**Note: This process will be repeated on each monitored server. This time, it is for the Main Server**.

1. Open the 04CreateAuditCopyJob.sql script in an SSMS query window with a connection to the Main Server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely not want to edit the DestShare or the AuditProxy values. The variables and default values from the script are:

:setvar SourceServer "ServerName"

:setvar SourceInstance "MSSQL"

:setvar SourcePath "C:\Audit"

:setvar MainServer "MainServerName"

:setvar RenamedSA "nosa"

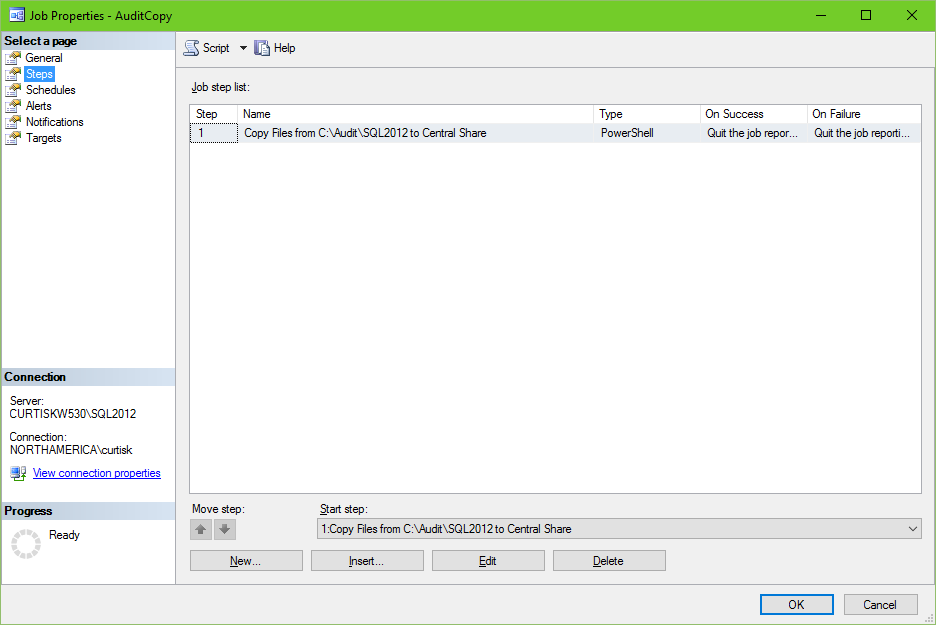
:setvar DestShare "SQLAudit"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query. (Note: If you have the wrong name for ‘sa’, you will get the following error: “Msg 14234, Level 16, State 1, Procedure sp\_verify\_job, Line 197

The specified '@owner\_login\_name' is invalid (valid values are returned by sp\_helplogins [excluding Windows NT groups]).”)

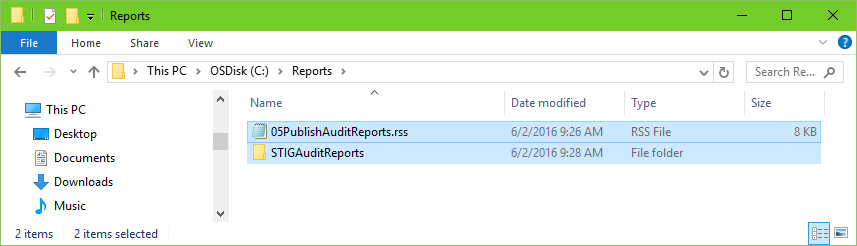
1. When finished, check for the existence of a SQL Agent job named “AuditCopy <Insstance>”
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “AuditCopy” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that a single step exists named “Copy files from ??? to ???” in the job. It should look something like the screenshot below.



### Publish the reports to the ReportServer. [Link](#_On_the_Central)

This section assumes that SQL Server Reporting Services has already been installed, configured and available. There are 24 pre-created SSRS reports that will display the Audit data. The 24 .rdl files are provided and the table in Appendix B describes them.

1. Extract the contents of STIGAuditReports.zip to a directory on the report server. We suggest creating a X:\Reports folder for this purpose.
2. Copy the 05PublishAuditReports.rss file to that same location.

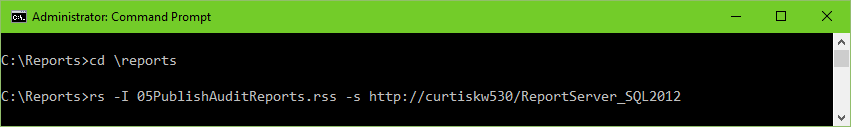


1. Edit the copy of 05PublishAuditReports.rss that was created in step 2 in Notepad.
2. Edit the variables in the script to match your environment. The rows that are shaded in the table below represent the only values that you will likely need to change. The settable variables and the given defaults/example values are as follows:

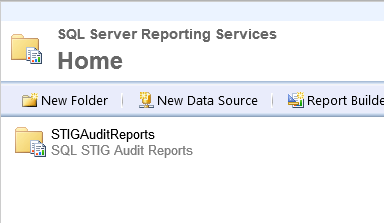
|  |  |  |
| --- | --- | --- |
| Variable | Default/Example | Description |
| SvrInst | "DBServer\Instance" | The Main Server and SQL Instance (Server name only for default instance) |
| rptPath | "c:\reports\STIGAuditReports" | Location of the .rdl files to be imported. |
| PrjDB | "SQLAudit" | Audit database name. There is no need to change this value. |
| DataSrcName | "SQLAuditDS" | The Name of the Shared Data Source for the reports. If you change this, you’ll have to edit this value for every report. I wouldn’t. |
| DataSrcDscr | "SQLAudit Datasource" | A description – No need to change. |
| parentFolder | "STIGAuditReports" | Folder in SSRS where the reports will be stored. This is not the file folder created in step 2 above. Suggest you not change this value. |
| parentFolderDscr | "SQL STIG Audit Reports" | Description for the parent folder above. No need to change this value. |

1. Save the edited file.
2. Open a command prompt “as Administrator”
3. Change the default directory to the location of the edited .rss file using cd \PathToTheRSSFile
4. Enter the following command line to publish the audit reports to the report server. Replace <http://servername/ReportServer> with the URL of your report server, probably only replacing the “servername” and adding the “\_InstanceName to the very end. The report server’s URL is available from the Report Server Configuration Manager. The command line is:

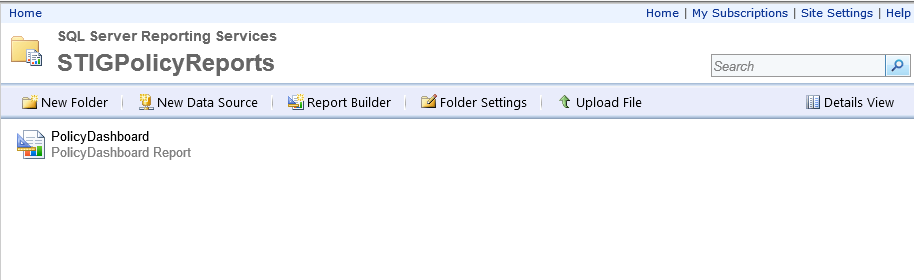
*rs -I 05PublishAuditReports.rss -s* [*http://servername/ReportServer*](http://servername/ReportServer)*\_Instance -eMgmt2010*



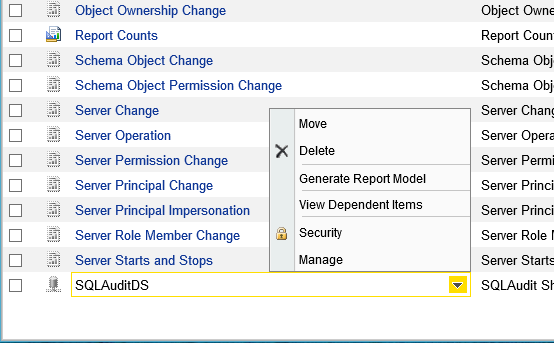
1. These reports have a dependency on other objects that are not installed as yet. We will confirm that the reports have installed correctly later in the process. For now, you can open an Internet Explorer browser ‘as Administrator’, to a link like <http://ServerName/Reports> (default instance) or <http://ServerName/Reports_InstanceName> (named instance). The Edge browser is not compatible with SSRS 2012. From there you should see a folder named STIGAuditReports.



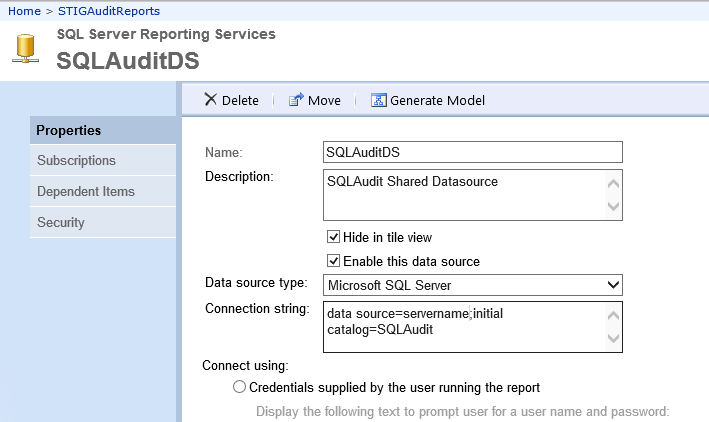
1. Open that folder (click on the folder name) and then click on the “Details View” control on the right side of the blue bar to reveal 24 reports and one shared datasource.



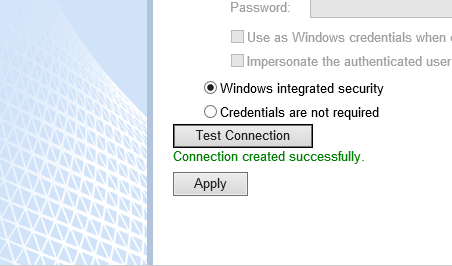
1. Click on the “Details View” link on the upper left under the Search control (See above screenshot for the “Details View” link location). The result should look like the screenshot below (shown with the dropdown menu from the next step).



1. Mouse-over “SQLAuditDS”, click on the drop-down arrow and choose Manage from the dropdown menu. The result should look like the screenshot below.



1. Change the name of the “data source” in the Connection string box. Above, the name is “servername”. Do not leave any spaces in the connection string. Optionally, you can click the “Test Connection” button at the bottom of the screen. A successful connection will show you the “Connection created successfully” message below:



1. Click the “Apply” button. Changes are not saved unless you click this button. It is recommended that you close and reopen this page to confirm that your change was saved.
2. Click the “Home” link at the top of the Report Manager to return to the Home screen.

### Set the permissions to view Audit information through the included reports. [Link](#_On_the_Central)

Test to find out what might need to go in this section.

## Audit Setup on each Monitored Server Instance

The steps in this section describe the actions to take to add a server\Instance to the servers monitored by the system. This assumes that the server and instance was added to the “Import Audit Files” job when it was created. If not, follow the optional steps below to add those steps to the job.

All of these steps can be performed from a remote server (‘remote’ means SQL Instances other than the Main Server including another instance on the same Windows server as the Main) so long as a sysadmin privileged SSMS query window can be opened to the remote server.

### Create the SQL instance Audit, the instance Audit specification, and Database Audit specifications, Audit Credential, and Audit Proxy. (02CreateAuditObjects.sql) [Link](#_On_each_server)

1. Open the 02CreateAuditObjects.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the files location for the files of the audit (AuditFilePath) and the Audit Credential Account (AuditCredAcct). The variables and default values from the script are:

:setvar AuditCredAcct "DOMAIN\UserName"

:setvar AuditCredPW "PasswordHere"

-- Recommend using wrong PW here and changing in UI (more secure)

-- Probably no need to change the values below

:setvar AuditFilePath "C:\Audit"

:setvar InstanceName "InstanceName"

:setvar IsMirrored "False"

:setvar MaxSize 100

:setvar MaxRolloverFiles 255

:setvar ReserveDiskSpace "OFF"

:setvar QueueDelay 1000

:setvar OnFailure CONTINUE

:setvar EnabledState ON

:setvar AuditCredObj "AuditCredential"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query. (You may get some errors if this instance of SQL Server does not support Database Audits. This is not required for the STIGs.)
3. When finished, check for the existence of the instance audit, instance audit specification, and the database audit. specifications.
   * In the SSMS Object Explorer, refresh the Security node.
   * In the Security node, expand the Audits node and confirm the existence of the audit named STIG\_COMPLIANT\_SERVER\_AUDIT.
   * In the Server Audit Specifications node, expand the node and confirm the existence of the audit specification named STIG\_COMPLIANT\_SERVER\_AUDIT
   * (Enterprise Edition only) In any of the user database, expand the database node, the Security node, and the Database Audit Specification node to confirm the existence of an audit specification named STIG\_COMPLIANT\_DATABASE\_AUDIT
   * In the SSMS Object Explorer, refresh the Security node and expand the Credentials node to confirm the existence of the “AuditCredential”. If you entered an incorrect password for the credential in the creation script, open the credential’s properties dialog and change the password.
   * In the SSMS Object Explorer, refresh the SQL Agent node and expand the SQL Agent and Proxies nodes for Operating System (CmdExec) and PowerShell to confirm the existence of the “AuditProxy”.

### Create the AuditCopy job that copies the audit data from the folder where the files are created to the correct folder on the Main Server. (04CreateAuditCopyJob.sql) [Link](#_On_each_server)

1. If the target folder does not already exist on the Main Server, go create it now.
   * On the Main Server, navigate to the storage folder for audit files. By default, this is the SQLAudit folder.
   * Create a subfolder named Server\_Instance where ‘Server’ is the name of the source server and ‘Instance’ is the name of the source instance. This folder will be the target for the AuditCopy job.
2. Open the 04CreateAuditCopyJob.sql script in an SSMS query window with a connection to the remote server.
3. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely not want to edit the DestShare or the AuditProxy values. The variables and default values from the script are:

:setvar SourceServer "ServerName"

:setvar SourceInstance "MSSQL"

:setvar SourcePath "C:\Audit"

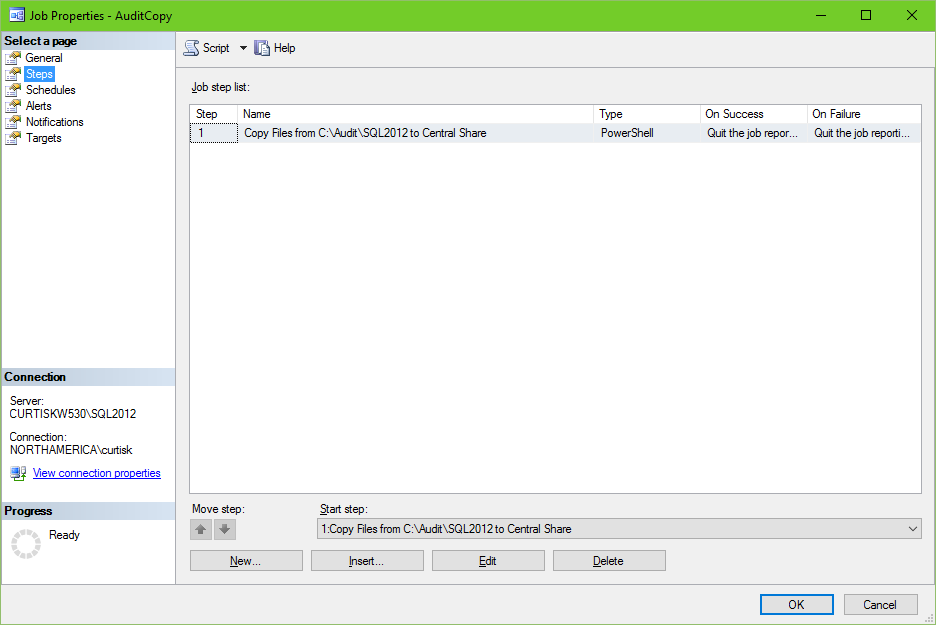
:setvar MainServer "ServerName"

:setvar RenamedSA "nosa"

:setvar DestShare "SQLAudit"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of a SQL Agent job on the remote server named “AuditCopy”
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “AuditCopy” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that a single step exists named “Copy files from ??? to ???” in the job. It should look something like the screenshot below.



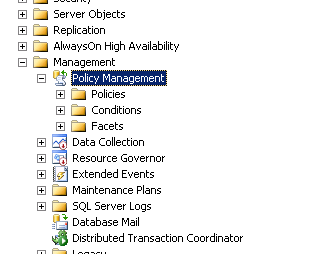
## Policy Management Setup

All steps are performed on the Main Server only, except where specifically marked.

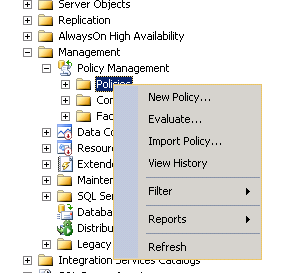
### Manually Import Policies (SQL2012v10 STIG Policies.zip) [Link](#_Policy_Section)

The ImportPolicies.ps1 script can be used to automate this step. Instructions for this PowerShell method are not included in this version.

1. Unzip the files
   * Copy the SQL2012v10 STIG Policies.zip file to a folder on the Main Server. ‘X:\Reports’ can be reused for this purpose or you can create a new folder.
   * Extract all files to that folder by right-clicking on the .zip file and choosing “Extract All…”
   * Make a note of that location. It will be needed in the next step when you import the policies.
2. Import the policies
   * Connect the Main instance in SSMS Object Explorer.
   * Open the ‘Management’ node
   * Open the ‘Policy Management’ node.

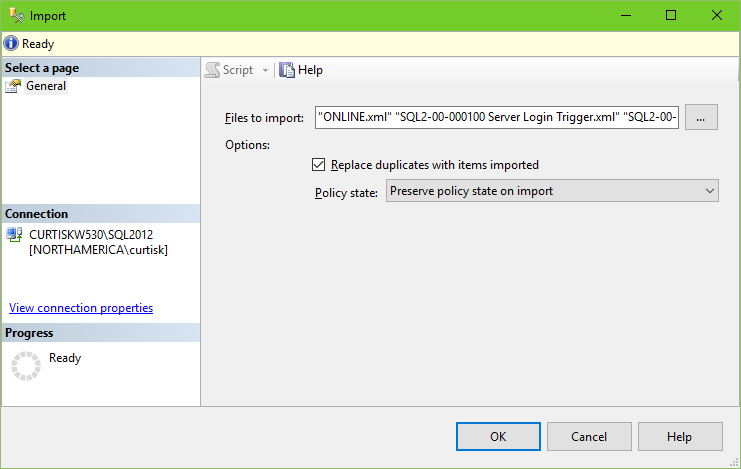


* + Right click on the ‘Policies’ node and choose ‘Import Policy…’ from the context menu.



The ‘Import’ window will pop up.

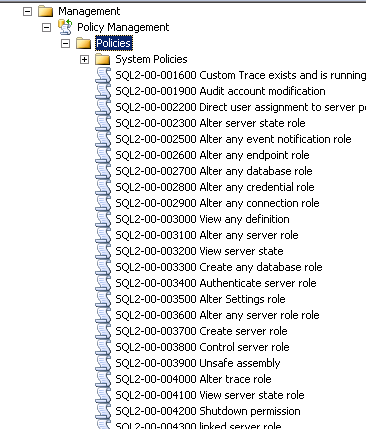
* + Click the ‘…’ button to the right of the ‘Files to import:’ textbox. The ‘Select Policy’ box will open up.
  + Browse to the folder where you unzipped the policies.
  + Select all of the policies by selecting a file then clicking CTRL-A.
  + Click the ‘Open’ button to select the policies and return to the “Import Policies” dialog.
  + Check the ‘Replace duplicates with items imported’ checkbox.



* + Click ‘OK’ to import the policies.

The policies will import into the ‘Policies’ folder in the SSMS Object Explorer.

* + Expand the Policies node in the Object Explorer of SSMS to confirm the import. You should see something like the screenshot below.



### Create the Database Objects and insert data (06CreatePolicyDB.sql) [Link](#_Policy_Section)

1. Open the 06CreatePolicyDB.sql script in an SSMS query window with a connection to the Main Server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the files location for the data and log file of the database. The variables and default values from the script are:

:setvar PolicyDBName "MDW"

:setvar PolicyDBDataFilePath "C:\{*path*}\DATA"

:setvar PolicyDBLogFilePath "C:\{*path*}\DATA"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the “MDW” database.
   * In the SSMS Object Explorer, refresh the databases folder.
   * Expand the list of databases.
   * Confirm the existence of the new “MDW” database.

### Configure the PowerShell user account rights [Link](#_Policy_Section)

1. Make note of the domain user account created for use in running the PowerShell scripts. [**Prerequisites**](#PS_Acct_Prereq)

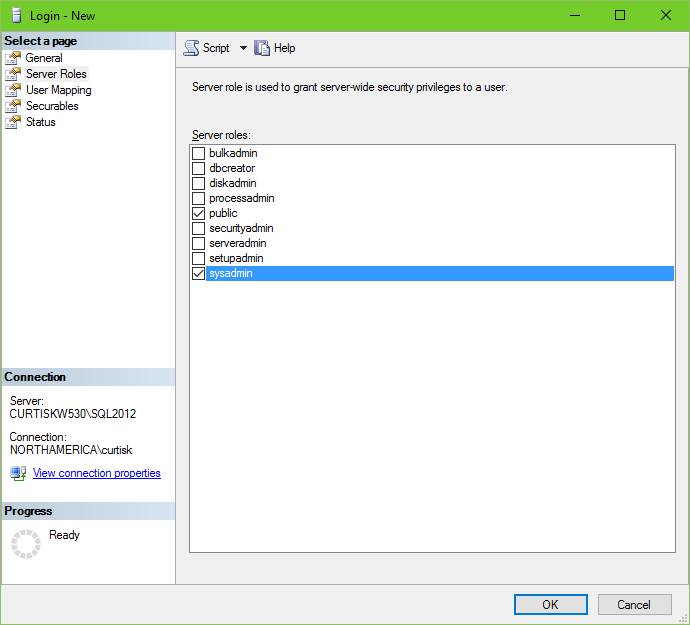
* Account needs domain user rights only.

1. Grant the user login rights

* Connect to the Main Server instance in SSMS Object Explorer.
* Expand the instance node, then the “Security” node.
* Right click on the “Logins” node and choose “New Login…” from the context menu.
* The “Login – New” box will pop up.
* In the “Login name:” textbox, type in the name of the PowerShell domain user account (Domain\UserAcct).

1. Grant the user sysadmin rights

* With the ‘Login’ dialog still open, click on the “Server Roles” under “Select a page”
* Check the box in front of “sysadmin”.



1. Grant the user permissions in master, msdb, and the PBM database.

* With the ‘Login’ dialog still open, click on “User Mapping” under “Select a page” on the left.
* Place a check in front of ‘master.
* In the “Database role membership for: master” pane place a check in front of the following roles:

db\_datareader

* Place a check in front of ‘msdb’.
* In the “Database role membership for: msdb” pane place a check in front of the following roles:

db\_datareader

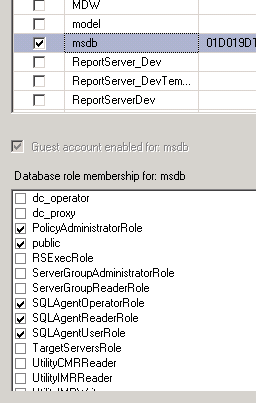
PolicyAdministratorRole

public

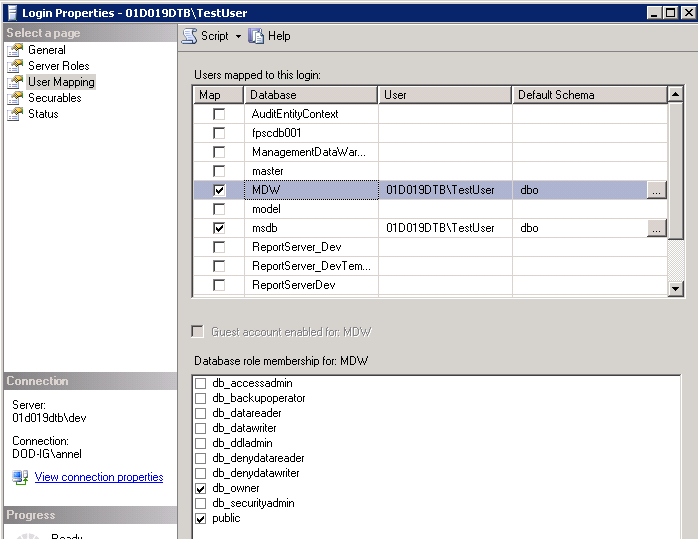
SQLAgentOperatorRole

SQLAgentReaderRole

SQLAgentUserRole



* Place a check in front of the PBM database. (In this case it is called MDW.)
* In the “Database role membership for: MDW” pane place a check in front of “db\_owner”.



* Click “OK” to accept

### Create Synonyms, Linked Server, Login, Credential, and Proxy (each monitored server) (07CreatePolicyObjects.sql) [Link](#_Policy_Section)

1. Open the 07CreatePolicyObjects.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will probably not want to edit the PolicyCredObj or PolicyProxy values. The variables and default values from the script are:

:setvar MainServer "ServerName"

:setvar MainInstName "MainInstanceName"

--use "" or "MSSQL" for default instances

:setvar PolicyCredAcct "Domain\LoginName"

--Domain account to be used for Credential/Proxy

:setvar PolicyCredPW "PasswordHere"

--Domain account PW to be used for PolicyCredAcct

-- Probably shouldn't change either of these

:setvar PolicyCredObj "PolicyCredential"

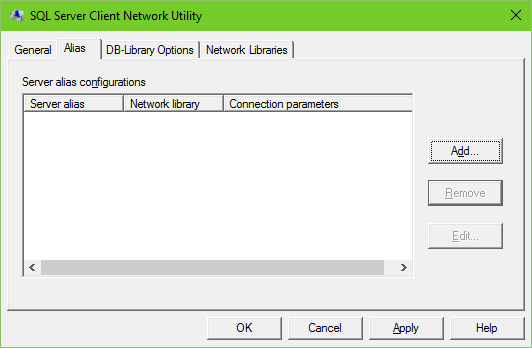
:setvar PolicyProxy "PolicyProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the credential and proxy.
   * In the SSMS Object Explorer, refresh the Security node and expand the Credentials node to confirm the existence of the “PolicyCredential.” If you entered the wrong password for PolicyCredPW in the scritp, go into the properties of the PolicyCredential object and enter the correct password.
   * In the SSMS Object Explorer, refresh the SQL Agent node and expand the SQL Agent and Proxies nodes to confirm the existence of the “PolicyProxy”.

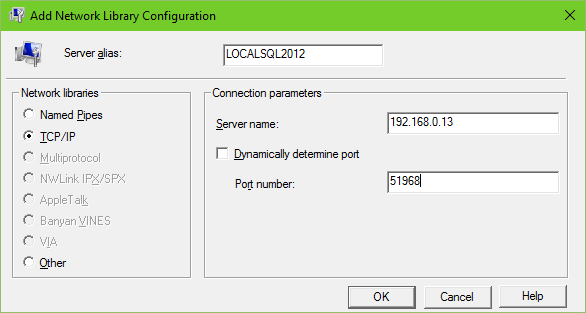
### Configure an Alias for the CMS Instance [Link](#_Policy_Section)

**Note: This step is only necessary for the CMS Instance of SQL Server. The CMS serverserverServer cannot evaluate policies against itself through a CMS registration. The alias gets around this limitation.**

1. From an Administrator Command Prompt (or hit Win+R), Type CLICONFG.exe
   * The SQL Server Client Network Utility dialog will appear.

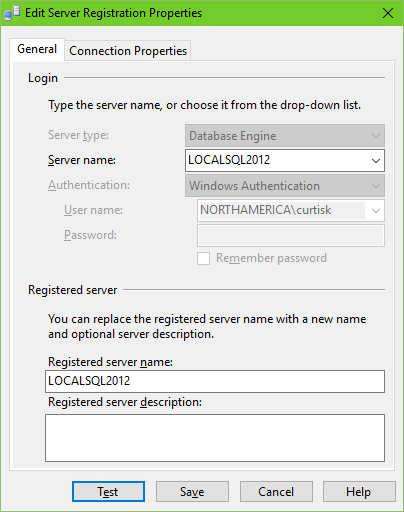


1. Click the “Add…” button.
   * The Add Network Library Configuration dialog will appear
2. Complete the dialog to create the Alias
   * Type the Alias Server Name in the Server Alias box
   * Choose TCP/IP under Network Libraries
   * Enter the Servers IP address for the “Server Name:”
   * Uncheck “Dynamically determine port”
   * Type the Instances Port Number into the “Port Number:” box



* + Click OK
  + The Alias should now appear in the SQL Server Client Network Utility.
  + Click OK to close the remaining dialog.

1. Add the server to a CMS Group
   * Right-click on a CMS Server Group under Registered Servers in SSMS and choose “New Server Registration”
   * For the Server Name, enter the Server’s Alias as created above.



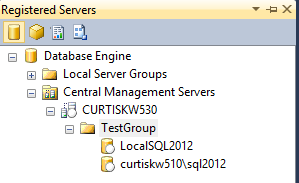
* + Click the “Test” button. You should get a dialog telling you the test was successful.
  + Click the “Save” button to close the dialog.

This registration will be the registration for the CMS instance.

### Manual policies test. [Link](#_Policy_Section)

1. Open up the Central Management Server in ‘Registered Servers’ in SSMS.

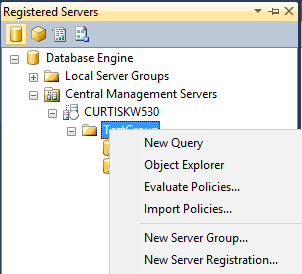
* Open a Group node as shown below.



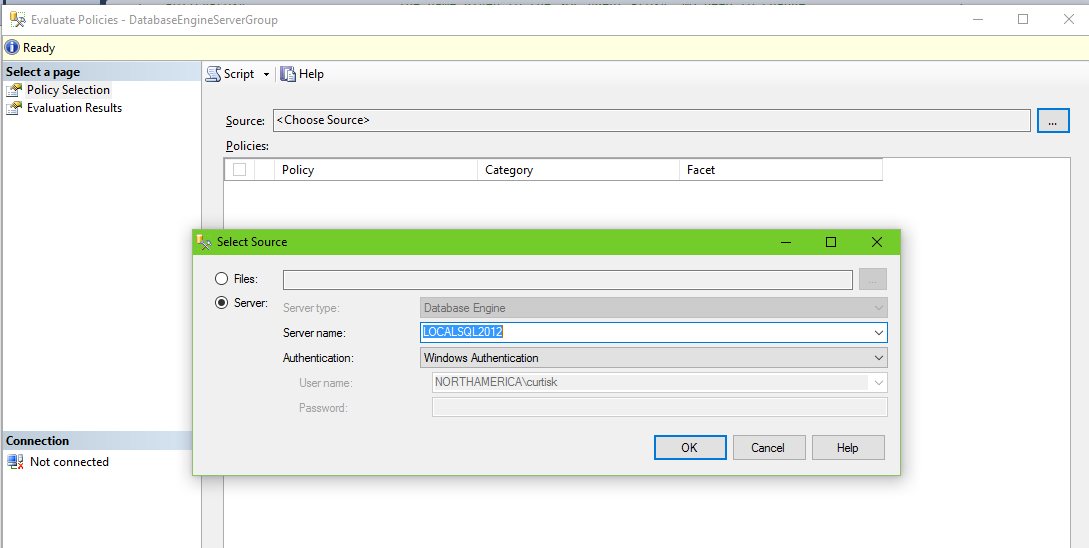
1. Evaluate Policies

* Right click on the group node and choose ‘Evaluate Policies’ from the context menu.

NOTE: Any group or sub-group can be chosen. Only the servers under that group will be included in the evaluation.

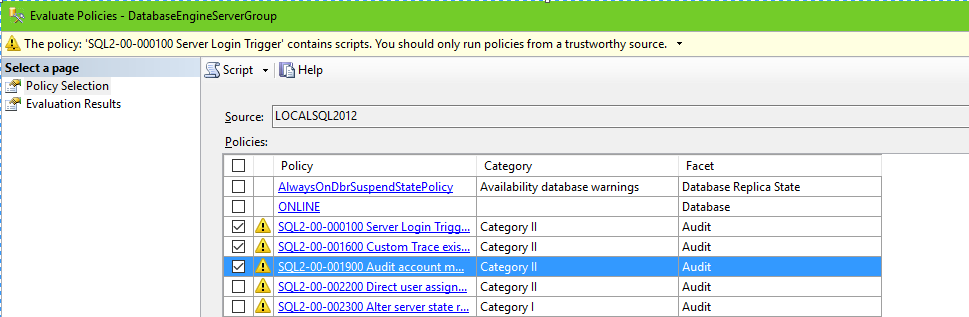


* The “Evaluate Policies - <group name>” box will pop up.
* Click the radio button to the right of “Source…”



* Choose the server node by selecting the “Server:” radio button.
* In the “Server name:” dropdown choose the name of the Main Server.
* Click the ‘OK’ button to accept.

The “Evaluate Policies - <group name> box will come back to the front.

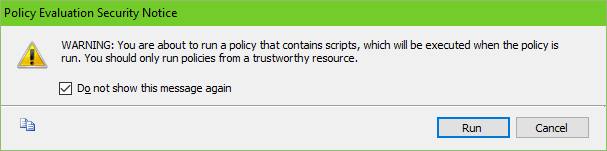


1. Evaluate the Policies

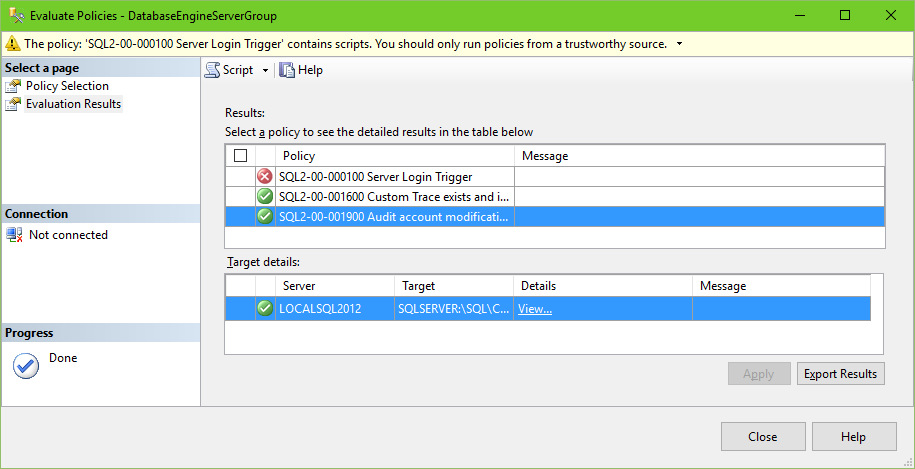
* In the ‘Evaluate Policies - <group name>’ box, check the box in front of the policies to be evaluated.

NOTE: If all policies are required, checking the box in the title bar will automatically check all of the policies.

* Click the ‘Evaluate’ button. You will get a warning that you are running policies that contains scripts. Click the box “Do not show this message again” and click “Run”.



* The results will appear in the ‘Evaluate Policies - <group name>’ box.



NOTE: Click on the ‘View’ hyperlink in the ‘Details’ column in the ‘Target details:’ section to view details of why the policy failed evaluation.

1. Export the results.

* Click the ‘Export Results’ button to export the results.

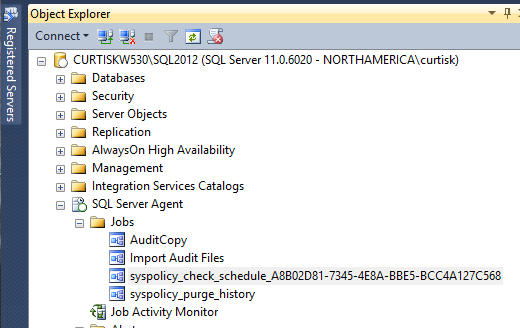
NOTE: Save the file to the file system with an ‘.xml’ extension. The results can be evaluated in EXCEL.

* Click “Save”.
* Evaluate the results, looking for errors in evaluation or finding the targets. Failure of the test (like finding an Audit Account Modification) is the desired result.

### Disable the auto-created evaluation job [Link](#_Policy_Section)

When importing the PBM Policies, a SQL Agent Job is automatically created to run those jobs on a schedule. We will replace this job below, so we need to disable this auto-created job.

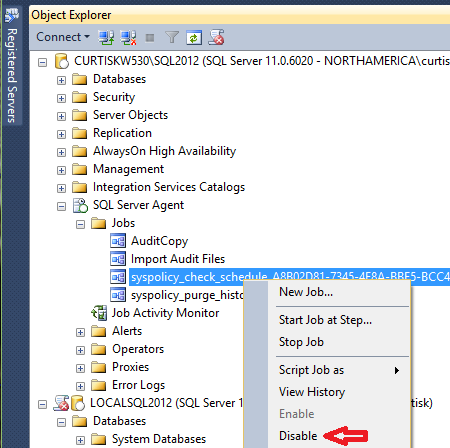
1. Expand the SQL Agent Node of the Main Server in the Object Explorer of SSMS



1. Right-click on the job whos name starts “syspolicy\_check\_schedule”

* If there is more than one, you will need to check the properties of each job to find the one that was created at the time of importing the policies.

1. From the context menus, choose “Disable”.



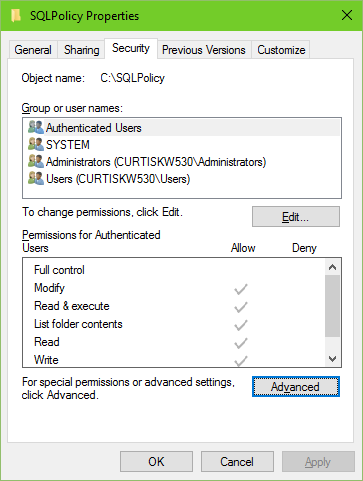
* The “Disable Jobs” dialog will appear.

1. Click “Close” to close the dialog.

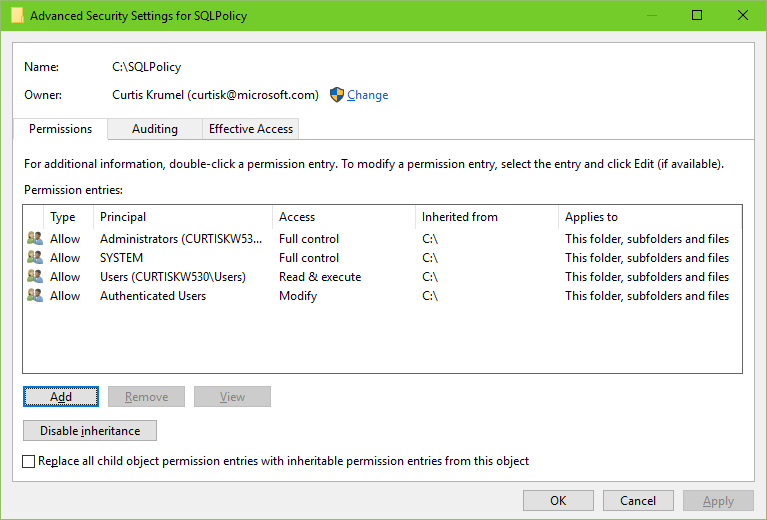
### Create the working directories [Link](#_Policy_Section)

These folders will be used by the **‘**Run Policies Check Job(s)’ during processing.

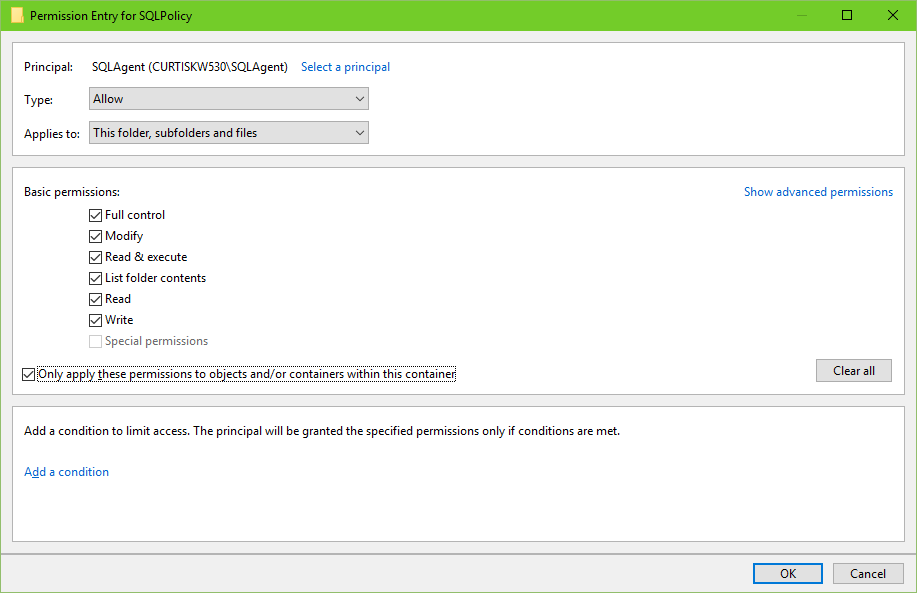
1. Open Windows Explorer by right clicking on the start button and choosing “Open Windows Explorer” from the context menu. Double-clicking on the “Computer” icon will work as well.
2. On the drive that will serve to temporarily hold the output of the PBM policies during job execution, create a folder. The scripts and these instructions name this folder “SQLPolicy,” so it is highly suggested that you name yours the same. These instructions will refer to this folder as X:\SQLPolicy where X: is any drive on the Main Server that has at least 1 GB of free space.
3. In X:\SQLPolicy, create a folder for each Instance group that will be evaluated by the **‘**Run Policies Check Job(s)’ (to be created shortly). This will likely be one folder for each server group (Server groups are the groups of servers by version number that you listed in the prerequisites. The groups will actually be created in the next section when the job to evaluate those groups is created.). Keep the names of the folders similar to the names of the groups (SQL2008R2, SQL2012, SQL2014 recommended). They will be used below as results directories (EvalResultsFolder )
4. Set the permissions of these folders such that the SQL Agent service account and the PowerShell user account have full control of these folders.
   * Right-click on the SQLPolicy folder and choose “Properties” from the context menu.
   * Click on the “Security” tab.



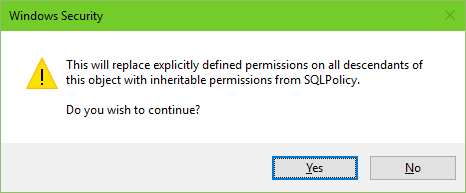
* + Click the “Advanced” button.
  + With the “Permissions” tab selected, click the “Add” button.



* + Do the following for each of the two accounts:
    1. Click “Select a Principal”
    2. Enter the account name in the box titled “Enter the object name to select” and click the “Check Name” button.
    3. Click OK
    4. Type: should read “Allow” and Applies to: should read “This folder, subfolders and files”. Leave them this way.
    5. Under Basic Permissions: check “Full Control” and “Only apply these permissions to object and/or containers within this container”.



* + 1. Click the OK button.
  + Once the permissions have been set for both accounts, on the Advanced Security Settings for SQLPolicy dialog, check the box beside “Replace all child object permission entries…”
  + Click the OK button
  + A Windows Security dialog will appear warning you about replacing permissions. Click “Yes” to continue.



* + Click the OK button on the SQLPolicy Properties dialog.

### Create the Import Policies Job [Link](#_Policy_Section)

1. Save the PowerShell script EvalPolicies.ps1 to a folder where it can be accessed. Recommend you save it as “X:\SQLPolicy” to match the default in these instructions.
2. Open the 08CreatePolicyImportJob.sql script in an SSMS query window with a connection to the Main Server.
3. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will probably not want to edit the PolicyDBName, JobCategory or JobName values. The variables and default values from the script are:

:setvar OwnerLogin "nosa"

:setvar EvalScriptPath "R:\SQLPolicy"

:setvar EvalScriptFile "EvalPolicies.ps1"

:setvar EvalGroupName "2012 SQL Servers"

:setvar EvalResultsFolder "2012SQLServers"

:setvar MainServer "ServerName"

:setvar MainInstName "InstanceName"

--use "" or "MSSQL" for default instances

-- Probably don't need to change values past here --

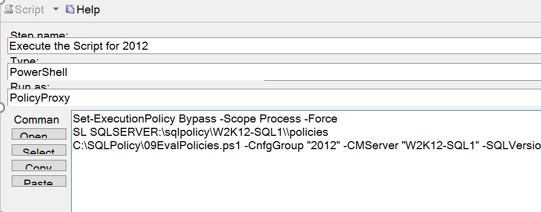
:setvar PolicyProxy "PolicyProxy"

:setvar PolicyDBName "MDW"

:setvar JobCategory "[Uncategorized (Local)]"

:setvar JobName "Import Policies"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query. This will create a job that evaluates all of the SQL instances in the specified Instance group.
3. When finished, check for the existence of the job(s).
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “Import Policies” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that a single step exists named “Execute the Script for <Instance group>” in the job.
   * Click the edit button to open the Job Step Properties. It should look something like the screenshot below.



1. To make other jobs steps that evaluate other Instance groups, change the EvalGroupName, EvalResultsFolder, and JobName values and re-execute the SQL script. Those jobs can be combined into one by copying the resulting commands (as shown in the screenshot above) into new steps of the central ‘Evaluate Policies’ job.

### Publish reports, data source, and shared dataset to SSRS (Report Server only) (09PublishPolicyReports.rss) [Link](#_Policy_Section)

1. Extract the contents of STIGPolicyReports.zip to a directory on the report server. We suggest creating/reusing ‘X:\Reports’ folder for this purpose.
2. Copy the 09PublishPolicyReports.rss file to that same location.
3. Open the 09PublishPolicyReports.rss file in Notepad.
4. Edit the variables in the script to match your environment. The rows that are shaded in the table below represent the only values that you will need to change. The settable variables and the given defaults/example values are as follows:

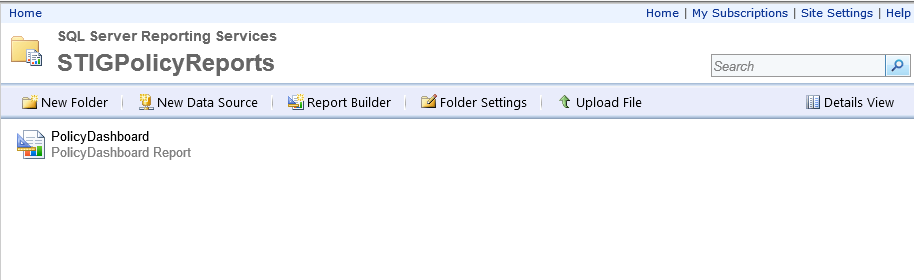
|  |  |  |
| --- | --- | --- |
| Variable | Default/Example | Description |
| SvrInst | "DBServer\Instance" | The Main Server and SQL Instance |
| PrjDB | "MDW" | Audit database name. There is no need to change this value. |
| DataSrcDscr | "SQLPolicy Datasource" | A description – No need to change. |
| RSUserid | "Domain\RSCredential" | Accont for running reports against the audit database. |
| RSPassword | "ChangeAfterInstallBR549!" | The password for the above account. Best to use the wrong password here and change it in the Report Manager later. |
| parentFolder | "STIGPolicyReports" | Folder in SSRS where the reports will be stored. Suggest you not change this value. |
| parentFolderDscr | "SQL STIG Policy Reports" | Description for the folder above. No need to change this value. |
| filePath | "C:\LocationOfRDLFiles" | Location of the .rdl files to be imported. |

1. Save the edited file.
2. Open a command prompt “as Administrator”
3. Change the default directory to the location of the edited .rss file using cd \PathToTheRSSFile. A good location would be X:\SQLPolicy where X: is the drive used for the Policy working directory above.
4. Enter the following command line to publish the audit reports to the report server. Replace <http://servername/ReportServer> with the URL of your report server, probably only replacing the “servername.” The report server’s URL is available from the Report Server Configuration Manager. The command line is:

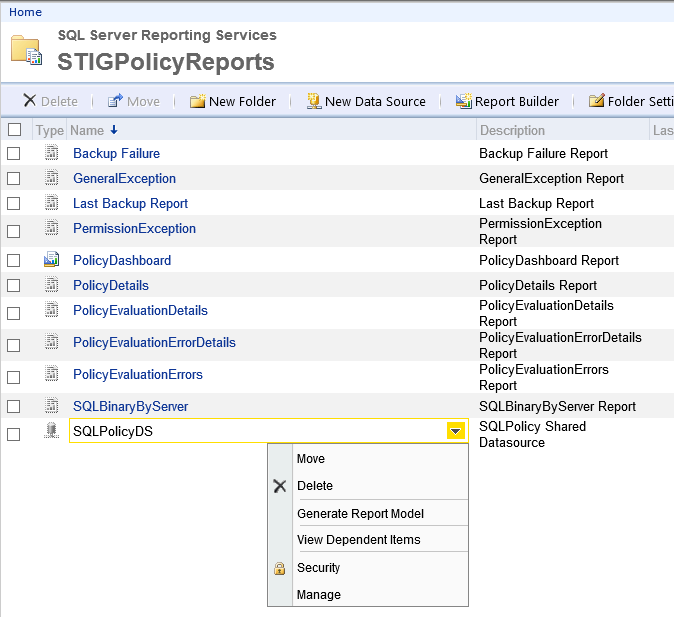
*rs -I 09PublishPolicyReports.rss -s* [*http://servername/ReportServer*](http://servername/ReportServer)*\_InstanceName -eMgmt2010*

*<screen shot needed here>*

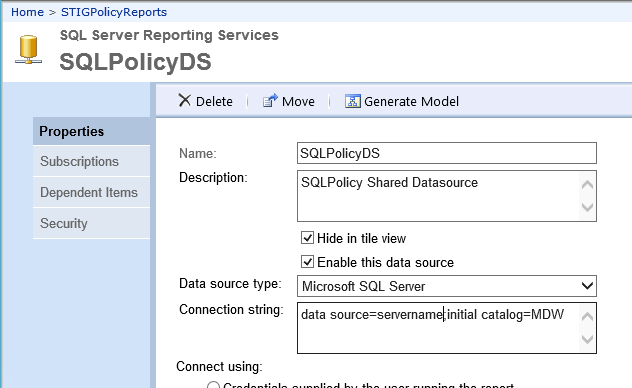
1. Confirm that the reports were published correctly by opening a browser to the URL of the Report Manager (<http://servername/reports>) and confirm that the folder specified in the parentFolder variable above exists (STIGPolicyReports by default).
2. Click on the folder to confirm that only the PolicyDashboard report is visible.



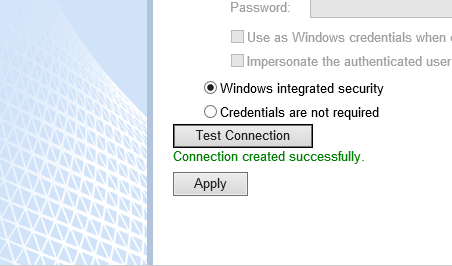
1. Click on the “Details View” link on the upper left under the Search control (See above screenshot for the “Details View” link location). The result should look like the screenshot below (shown with the dropdown menu from the next step).



1. Mouse-over “SQLPolicyDS”, click on the drop-down arrow and choose Manage from the dropdown menu. The result should look like the screenshot below.



1. Change the name of the “data source” in the Connection string box. Above, the name is “servername”. Do not leave any spaces in the connection string. Optionally, you can click the “Test Connection” button at the bottom of the screen. A successful connection will show you the “Connection created successfully” message below:



1. Click the “Apply” button. Changes are not saved unless you click this button. It is recommended that you close and reopen this page to confirm that your change was saved.
2. Click the “Home” link at the top of the Report Manager to return to the Home screen.

# Adding a new Server Group to the System for Evaluation

Follow these steps to add a new Server Group to the policy evaluation on your Central Management Server. These instructions assume that the Server Group has already been created in the database.

### Create a new folder under X:\SQLPolicy for the new Server Group

1. In X:\SQLPolicy, create a folder for the new Instance group. Keep the name of the folder similar to the name of the server group. It will be used as results directories (EvalResultsFolder or -ResultDir) for policy evaluation.
2. Set the permissions of this folder such that the SQL Agent service account and the PowerShell user account have full control of these folders.

### Either add a step to the Import Policy Job or create a new Import Policy job

To add a new Import Policy Job:

1. Open the 10CreatePolicyImport Job.sql script in an SSMS query window with a connection to the Main Server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will probably not want to edit the PolicyDBName or JobCategory. JobName must be unique for among job names on the Main Server. The variables and default values from the script are:

:setvar OwnerLogin "sa"

:setvar EvalScriptPath "R:\SQLPolicy"

:setvar EvalScriptFile "EvalPolicies.ps1"

:setvar EvalGroupName "2012 SQL Servers"

:setvar EvalResultsFolder "2012SQLServers"

:setvar MainServer "ServerName"

:setvar MainInstName "SQL2012"

--use "" or "MSSQL" for default instances

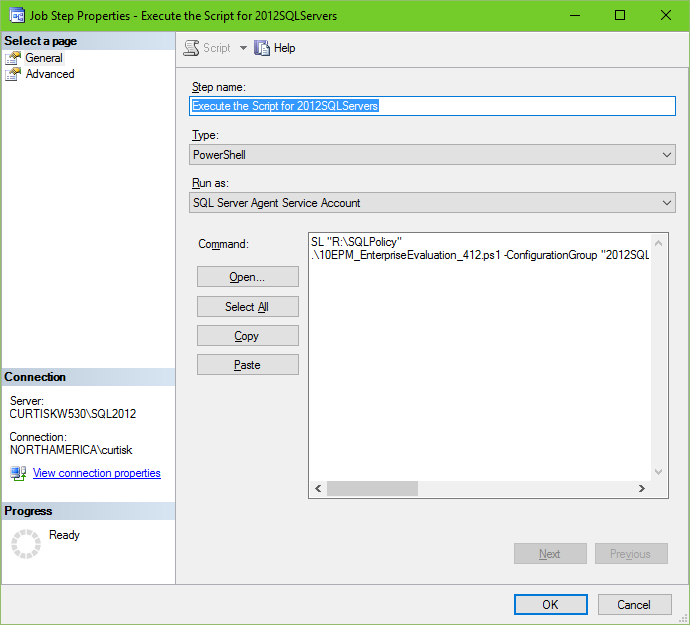
-- Probably don't need to change values past here --

:setvar PolicyDBName "MDW"

:setvar JobCategory "[Uncategorized (Local)]"

:setvar JobName "Import Policies"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query. This will create a job that evaluates all of the SQL instances in the specified Instance group.
3. When finished, check for the existence of the job(s).
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “Import Policies” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that a single step exists named “Execute the Script for ???” in the job.
   * Click the edit button to open the Job Step Properties. It should look something like the screenshot below.



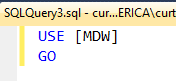
1. This job can be combined into a central Import Policies job by copying the resulting commands (as shown in the screenshot above) into new steps of the central job.

# Adding Instances to an Existing Monitored Group

This section is for adding SQL Server Instances to a STIG Monitoring system that has been installed and confirmed to be running normally.

### Go through the Checklist of Items Required for Monitored Instances.

1. Register the new server in the instance list using Policy.usp\_InsertServer.
   * Open a new query window connected to the Main Server.
   * Type in the new query window the following:



* + Execute the two rows then delete the lines.
  + In the same query window, type the following. Repeat for each new server to be monitored:

exec Policy.usp\_InsertServer [CurtisKW510\InstName], 'SQL2012'

where [CurtisKW510\InstName] is the server and instance name and 'SQL2012' is the group to which that instance belongs.

* + To list the contents of the Policy.Servers table (which determines the servers and instances and the groups to which they belong), execute the following:

EXEC [Policy].[usp\_ListServers]

1. Note the name and password of a domain user account for use in running SQL Agent Audit jobs.

SQL Audit Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Note the name and password of a domain user account for use in running the PowerShell scripts. Can be the same as the audit account above.

SQL PowerShell Account Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Local or SAN attached drive on the new server to store the audit files for the instance. 5 GB per monitored server is a suggested minimum for moderately busy servers.

Location for audit file storage: \_\_\_\_\_:/Audit

### Create the folders for the new instance’s audit files

1. On the Main Server using Windows Explorer, open the X:\SQLAudit folder.
2. Create a new folder with a name made of the ServerName\_InstanceName of the instances being added. (As before, use MSSQL for the InstanceName of a default instance.)
3. Assign Read and Delete permissions on this folder to the domain user account identified in step 5.1.2 above if these permissions do not already exist.
4. On the remote server using Windows Explorer, open the drive that was identified in step 5.1.4 (above). These instructions will refer to it as X:
5. Create a folder on this drive called “Audit”.
6. Create a sub-folder in X:\Audit with the name of the SQL Instance being added. (As before, use MSSQL for the Instance Name of a default instance.)
7. Assign Read and Delete permissions on the X:\Audit folder and its child folders to the domain user account identified in step 5.1.2 above if these permissions do not already exist.

All of the following steps can be performed from a any computer (like the Main Server) so long as a sysadmin privileged SSMS query window can be opened to the remote server.

### Create the SQL instance Audit, the instance Audit specification, and the Database Audit specification. (02CreateAuditObjects.sql) [Link](#_On_each_server)

1. Open the 02CreateAuditObjects.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the files location for the files of the audit (AuditFilePath). The variables and default values from the script are:

:setvar AuditFilePath "C:\Audit"

:setvar IsMirrored "False"

:setvar MaxSize 100

:setvar MaxRolloverFiles 255

:setvar ReserveDiskSpace "OFF"

:setvar QueueDelay 1000

:setvar OnFailure CONTINUE

:setvar EnabledState ON

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the instance audit, instance audit specification, and the database audit. specifications.
   * In the SSMS Object Explorer, refresh the Security node.
   * In the Security node, expand the Audits node and confirm the existence of the audit named STIG\_COMPLIANT\_SERVER\_AUDIT.
   * In the Server Audit Specifications node, expand the node and confirm the existence of the audit specification named STIG\_COMPLIANT\_SERVER\_AUDIT
   * (Enterprise Edition only) In any of the user database, expand the database node, the Security node, and the Database Audit Specification node to confirm the existence of an audit specification named STIG\_COMPLIANT\_DATABASE\_AUDIT

### Create the Proxy and Credential (02CreateAuditObjects.sql) [Link](#_On_each_server)

1. Open the 02CreateAuditObjects.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely only want to edit the AuditCredAcct and AuditCredPW. The variables and default values from the script are:

:setvar AuditCredAcct "DOMAIN\UserName"

:setvar AuditCredPW "PasswordHere"

-- Recommend using wrong PW here and changing in UI (more secure)

-- Probably no need to change the values below

:setvar AuditCredObj "AuditCredential"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the credential and proxy.
   * In the SSMS Object Explorer, refresh the Security node and expand the Credentials node to confirm the existence of the “AuditCredential”
   * In the SSMS Object Explorer, refresh the SQL Agent node and expand the SQL Agent and Proxies nodes to confirm the existence of the “AuditProxy”.

### Create the AuditCopy job that copies the audit data from the folder where the files are created to the correct folder on the Main Server. (04CreateAuditCopyJob.sql) [Link](#_On_each_server)

1. Open the 04CreateAuditCopyJob.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will likely not want to edit the DestShare or the AuditProxy values. The variables and default values from the script are:

:setvar SourceServer "ServerName"

:setvar SourceInstance "MSSQL"

:setvar SourcePath "C:\Audit"

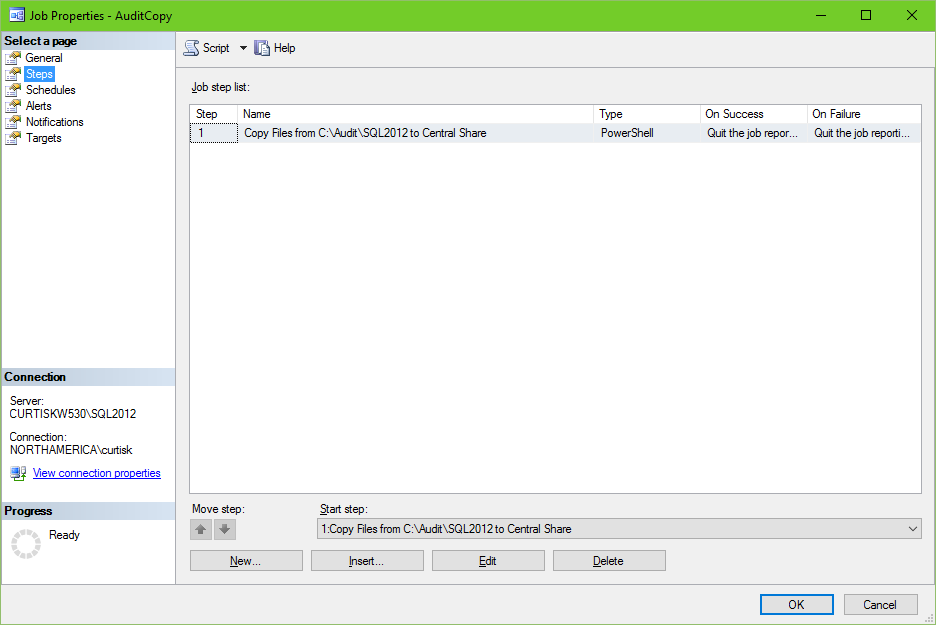
:setvar MainServer "ServerName"

:setvar RenamedSA "nosa"

:setvar DestShare "SQLAudit"

:setvar AuditProxy "AuditProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of a SQL Agent job on the remote server named “AuditCopy”
   * Refresh the SQL Agent node, then expand the SQL Agent and Jobs nodes to confirm the existence of the job.
   * Right-click on the “AuditCopy” job and choose Properties.
   * Click on the Steps page on the left of the dialog and confirm that a single step exists named “Copy files from ??? to ???” in the job. It should look something like the screenshot below.



### Edit the “Import Audit Files” job in the SQL Agent of the Main Server to include the new server.

Another option for this step (updating the “Import Audit Files”) would be to reopen the 03CreateAuditImportFilesjob.sql script, add the new instances to the list of INSERT statements and rerun that script. See section [4.1.6](https://microsoft-my.sharepoint.com/personal/curtisk_microsoft_com/Documents/SQL%20STIGs/AnneLabbe/Import#_Edit_the_) for instructions on running that script.

1. Add a step to copy the files into the working directory.

* Open the “Import Audit Files” job by double clicking on it under the SQL Agent node in SSMS.
* Click on the ‘Steps’ page and then the “New…” button to add a step.
* In the step name type “Copy files from <instance name> to <destination path>”, where <instance name is the name of the SQL instance. Use the existing steps as references.
* In the “Type:” dropdown box, choose “Operating System”.
* In the “Run as” dropdown, choose AuditProxy.
* In the “Command:” textbox type the following code:

N'robocopy "$(SourcePath)\ $(SourceServer)\_$(SourceInstance)" "\\$(MainServer)\$(DestShare)\$(SourceServer)\_$(SourceInstance)" \*.sqlaudit /mov',

Example: robocopy "C:\SQLAudit\Server2\_MSSQL" "C:\SQLAudit\Working" \*.sqlaudit /mov

* On the Advanced page of the Job Step Properties dialog, set both the “On success action” and the “On failure action” to “Go to the next step”.
* Click “OK” to accept.
* Move the newly created job step to just above the current last “Copy” job.

1. Create a step to Import the file data into the Audit database.

* Assuming the Job Properties/Steps page is still open from the previous step, Click the “New…” button to add a step.
* In the step name type “Import Existing Files for <Server\Instance>”, where < Server\Instance > is the name of the SQL server source instance.
* In the “Type:” dropdown box, choose “Transact-SQL script (T-SQL)”.
* In the “Run as” dropdown leave it blank.
* In the “Database:” drop down, choose the name of the Audit database (SQLAudit).
* In the “Command:” textbox type the following code:

--Execute the main stored procedure

--Instance xxxxxxxx\DEV

EXEC dbo.usp\_Main\_Import\_audit\_logs '<path of the SQL audit files>'

--Example: EXEC dbo.usp\_Main\_Import\_audit\_logs 'C:\SQLAudit\Working\\*'

* On the Advanced page of the Job Step Properties dialog, set both the “On success action” and the “On failure action” to “Go to the next step”.
* Click “OK” to accept.
* Move the newly created job step to just below the “Copy” job created above.

1. Create a step to Delete the file data that was just imported into the Audit database.

* Assuming the Job Properties/Steps page is still open from the previous step, Click the “New…” button to add a step.
* In the step name type “Delete Audit Files from <Server\Instance>”, where < Server\Instance > is the name of the SQL server source instance.
* In the “Type:” dropdown box, choose “PowerShell”.
* In the “Run as” dropdown choose AuditProxy.
* In the “Database:” drop down, choose the name of the Audit database (SQLAudit).
* In the “Command:” textbox type the following code:

Remove-Item ‘<path of the SQL audit files>\\*.sqlaudit'

Example: Remove-Item 'C:\SQLAudit\Working\\*.sqlaudit'

* On the Advanced page of the Job Step Properties dialog, set both the “On success action” and the “On failure action” to “Go to the next step”.
* Click “OK” to accept.
* Move the newly created job step to just below the “Import” job created above.

### Create Synonyms, Linked Server, Login, Credential, and PolicyProxy (07CreatePolicyObjects.sql)

1. Open the 07CreatePolicyObjects.sql script in an SSMS query window with a connection to the remote server.
2. Set the script variables to the desired values as noted in the [table](#_Running_the_Scripts) above. You will probably not want to edit the PolicyCredObj or PolicyProxy values. The variables and default values from the script are:

:setvar MainServer "ServerName"

:setvar MainInstName "InstanceName"

--use "" or "MSSQL" for default instances

:setvar PolicyCredAcct "Domain\LoginName"

--Domain account to be used for Credential/Proxy

:setvar PolicyCredPW "PasswordHere"

--Domain account PW to be used for PolicyCredAcct

:setvar PolicyCredObj "PolicyCredential"

:setvar PolicyProxy "PolicyProxy"

1. Set the script to run in SQLCMD mode
   * With the focus on the query window, select the ‘Query’ menu from the top of the SSMS.
   * The SQLCMD selection will be third from the bottom. Click it. The background of the variables will turn to dark gray.
2. Execute the Query.
3. When finished, check for the existence of the credential and proxy.
   * In the SSMS Object Explorer, refresh the Security node and expand the Credentials node to confirm the existence of the “PolicyCredential”
   * In the SSMS Object Explorer, refresh the SQL Agent node and expand the SQL Agent and Proxies nodes to confirm the existence of the “PolicyProxy”.

# Appendix A – Reports

The 24 .rdl files along with their source stored procedure and SQL table are listed below:

|  |  |  |  |
| --- | --- | --- | --- |
| Report File | Stored Procedure | Table | Description from usp\_Main\_Import\_audit\_logs |
| Application Role Password Change.rdl | usp\_Merge\_audit\_application\_role\_password\_change | Audit\_application\_role\_password\_change | 13.) Merge data from the temporary import table into the Audit\_application\_role\_password\_change table |
| Backup and Restore.rdl | usp\_Merge\_audit\_backup\_restore | Audit\_backup\_restore | 15.) Merge data from the temporary import table into the Audit\_backup\_restore table |
| Server Change.rdl | usp\_Merge\_audit\_change | Audit\_change | 17.) Merge data from the temporary import table into the Audit\_change table |
| Database Change.rdl | usp\_Merge\_audit\_database\_change | Audit\_database\_change | 18.) Merge data from the temporary import table into the Audit\_database\_change table |
| Permission Change.rdl | usp\_Merge\_audit\_database\_permission\_change | Audit\_database\_permission\_change | 5.) Merge data from the temporary import table into the Audit\_database\_permission\_change table |
| Database Principal Change.rdl | usp\_Merge\_audit\_database\_principal\_change | Audit\_database\_principal\_change | 11.) Merge data from the temporary import table into the Audit\_database\_principal\_change table |
| Database Role Member Change.rdl | usp\_Merge\_audit\_database\_role\_member\_change | Audit\_database\_role\_member\_change | 12.) Merge data from the temporary import table into the Audit\_database\_role\_member\_change table |
| DBCC.rdl | usp\_Merge\_DBCC | Audit\_DBCC | 16.) Merge data from the temporary import table into the Audit\_DBCC table |
| Failed Login.rdl | usp\_Merge\_audit\_failed\_login | Audit\_failed\_login | 4.) Merge data from the temporary import table into the Audit\_failed\_login table |
| Login.rdl | usp\_Merge\_audit\_login | Audit\_Login | 1.) Merge data from the temporary import table into the Audit\_Login table |
| Login Change Password.rdl | usp\_Merge\_audit\_login\_change\_password | Audit\_login\_change\_password | 9.) Merge data from the temporary import table into the Audit\_login\_change\_password table |
| Logout.rdl | usp\_Merge\_audit\_logout | Audit\_Logout | 2.) Merge data from the temporary import table into the Audit\_Logout table |
| Object Change.rdl | usp\_Merge\_audit\_object\_change | Audit\_object\_change | 22.) Merge data from the temporary import table into the Audit\_object\_change |
| Object Ownership Change.rdl | usp\_Merge\_audit\_ownership\_change | Audit\_object\_ownership\_change | 21.) Merge data from the temporary import table into the Audit\_object\_ownership\_change |
| Schema Object Change.rdl | usp\_Merge\_audit\_schema\_object\_change | Audit\_schema\_object\_change | 14.) Merge data from the temporary import table into the Audit\_schema\_object\_change table |
| Schema Object Permission Change.rdl | usp\_Merge\_audit\_schema\_object\_permission\_change | Audit\_schema\_object\_permission\_change | 6.) Merge data from the temporary import table into the Audit\_schema\_object\_permission\_change table |
| Server Operation.rdl | usp\_Merge\_audit\_server\_operation | Audit\_server\_operation | 23.) Merge data from the temporary import table into the Audit\_server\_operation |
| Server Permission Change.rdl | usp\_Merge\_audit\_server\_permission\_change | Audit\_server\_permission\_change | 8.) Merge data from the temporary import table into the Audit\_server\_permission\_change table |
| Server Principal Change.rdl | usp\_Merge\_audit\_server\_principal\_change | Audit\_server\_principal\_change | 7.) Merge data from the temporary import table into the Audit\_server\_principal\_change table |
| Server Principal Impersonation.rdl | usp\_Merge\_audit\_server\_principal\_impersonation | Audit\_server\_principal\_impersonation | 20.) Merge data from the temporary import table into the Audit\_server\_principal\_impersonation |
| Server Role Member Change.rdl | usp\_Merge\_audit\_server\_role\_member\_change | Audit\_server\_role\_member\_change | 10.) Merge data from the temporary import table into the Audit\_server\_role\_member\_change table |
| Server Starts and Stops.rdl | usp\_Merge\_audit\_start\_stop | Audit\_server\_starts\_and\_stops | 3.) Merge data from the temporary import table into the Audit\_server\_starts\_and\_stops table |
| Audit Trace Change.rdl | usp\_Merge\_audit\_trace\_change | Audit\_trace\_change | 19.) Merge data from the temporary import table into the Audit\_trace\_change table |
| Database Permission Change | Usp\_Merge\_audit\_database\_permission\_change | Audit\_database\_permission\_change | 20.) Merge data from the temporary import table into the Audit\_database\_permission\_change table |

# Appendix B – Auditing for STIGs Detail

The SQL Server 2012 DISA STIG requires auditing to resolve potential vulnerabilities. The mechanism that the STIG specifies as a resolution for these findings is to create a SQL trace and include the events and columns specified in a SQL Trace script. The trace script creates an audit that DISA identifies as satisfactory for compliance. All scripts for SOCOM to be in compliance with DISA will be included with this step-by-step documentation on how to use them.

Microsoft released the SQL Server Audit feature in the SQL Server 2008 version, so it is now a very mature and incredibly reliable feature. SQL Server Audit uses Extended Events to create an audit file. For more information about SQL Server Audit and how it relates to Extended Events, see <http://msdn.microsoft.com/en-us/library/cc280386.aspx>.

This document is Step 2 in the overall solution that Microsoft is providing to SOCOM to help the command drive towards achieving STIG compliance for SQL Server. As a prerequisite, the reader should have an understanding of the STIG requirement and a basic understanding of the SQL Audit architecture and how SQL Audit works.

By the conclusion of the steps contained in this document, each server that has been identified as a server to be audited, will have a working Server Audit Specification and a working Database Audit Specification.



* 1. Required Trace Items

According to STIG ID: SQL2-00-011200, Rule ID: SV-53928r3\_rule, when using SQL Trace, the STIG requires the following items be included in a SQL Server audit. All objects will be provided in the form of T-SQL scripts and configuration guidance.

|  |  |  |
| --- | --- | --- |
| Trace ID | Trace Name | Trace Description |
| 14 | Audit Login | Occurs when a user successfully logs in to SQL Server |
| 15 | Audit Logout | Occurs when a user logs out of SQL Server |
| 18 | Audit Server Start & Stop | Occurs when the SQL Server service state is modified |
| 20 | Audit Login failed | Indicates that a login attempt to SQL Server from a client failed |
| 102 | Audit Database Scope GDR | Occurs every time a GRANT, DENY, REVOKE for a statement permission is issued by any user in SQL Server for database-only actions such as granting permissions on a database. |
| 103 | Audit Object GDR | Occurs every time a GRANT, DENY, REVOKE for an object permission is issued by any user in SQL Server |
| 104 | Audit Add login | Occurs when a SQL Server login is added or removed; for **sp\_addlogin** and **sp\_droplogin** |
| 105 | Audit login GDR | Occurs when a Windows login right is added or removed; for sp\_grantlogin, sp\_revokelogin, and sp\_denylogin |
| 106 | Audit login change | |  |  | | --- | --- | |  |  |   Occurs when a property of a login, except passwords, is modified; for **sp\_defaultdb** and **sp\_defaultlanguage** |
| 107 | Audit login change password | Occurs when a SQL Server login password is changed. Passwords are not recorded |
| 108 | Audit Add Login to Server Role | Occurs when a login is added or removed from a fixed server role; for **sp\_addsrvrolemember**, and **sp\_dropsrvrolemember** |
| 109 | Audit Add DB User | Occurs when a login is added or removed as a database user (Windows or SQL Server) to a database; for **sp\_grantdbaccess**, **sp\_revokedbaccess**, **sp\_adduser**, and **sp\_dropuser** |
| 110 | Audit Add Member to DB Role Event | Occurs when a login is added or removed as a database user (fixed or user-defined) to a database; for **sp\_addrolemember**, **sp\_droprolemember**, and **sp\_changegroup** |
| 111 | Audit Add Role | Occurs when a login is added or removed as a database user to a database; for **sp\_addrole** and **sp\_droprole** |
| 112 | Audit App Role Change Event | Occurs when a password of an application role is changed |
| 113 | Audit Statement Permission Event | Occurs when a statement permission (such as CREATE TABLE) is used |
| 115 | Audit backup/Restore Event | Occurs when a BACKUP or RESTORE command is issued |
| 116 | Audit DBCC Event | Occurs when DBCC commands are issued |
| 117 | Audit Change Audit Event | Occurs when audit trace modifications are made |
| 118 | Audit Object Derived Permission Event | Occurs when a CREATE, ALTER, and DROP object commands are issued |
| 128 | Audit database management event | Occurs when a database is created, altered, or dropped |
| 129 | Audit database object management event | Occurs when a CREATE, ALTER, or DROP statement executes on database objects, such as schemas |
| 130 | Audit Database Principal Management Event | Occurs when principals, such as users, are created, altered, or dropped from a database |
| 131 | Audit Schema Object Management Event | Occurs when server objects are created, altered, or dropped |
| 132 | Audit server principal impersonation event | Occurs when there is an impersonation within server scope, such as EXECUTE AS LOGIN |
| 133 | Audit database principle impersonation event | Occurs when an impersonation occurs within the database scope, such as EXECUTE AS USER or SETUSER |
| 134 | Audit server object take ownership event | Occurs when the owner is changed for objects in server scope |
| 135 | Audit Database object take ownership event | Occurs when a change of owner for objects within database scope occurs |
| 152 | Audit Change Database Owner | Occurs when ALTER AUTHORIZATION is used to change the owner of a database and permissions are checked to do that |
| 153 | Audit Schema Object Take Ownership Event | Occurs when ALTER AUTHORIZATION is used to assign an owner to an object and permissions are checked to do that |
| 170 | Audit Server Scope GDR | Indicates that a grant, deny, or revoke event for permissions in server scope occurred, such as creating a login |
| 171 | Audit Server Object GDR | Indicates that a grant, deny, or revoke event for a schema object, such as a table or function, occurred |
| 172 | Audit Database Object GDR | Indicates that a grant, deny, or revoke event for database objects, such as assemblies and schemas, occurred |
| 173 | Audit Server Operation Event | Occurs when Security Audit operations such as altering settings, resources, external access, or authorization are used |
| 175 | Audit Server Alter Trace Event | Occurs when a statement checks for the ALTER TRACE permission |
| 176 | Audit Server Object Management Event | Occurs when server objects are created, altered, or dropped |
| 177 | Audit server principal management event | Occurs when server principals are created, altered, or dropped. |
| 178 | Audit Database Operation Event | Occurs when database operations occur, such as checkpoint or subscribe query notification |

* 1. Trace Event ID mapping SQL Audit Action Group

The events included in an audit are not specified by event ID. In a SQL Server Audit, specify events by “Action Groups”. The Action Groups do not correspond directly to event IDs; however, the information included in an Action Group is very similar to the information included in a SQL Audit.

The following table contains the mappings from the SQL Trace ID to the SQL Audit Action Group that satisfies the trace.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trace ID | Trace Description | Audit Action Group | Server Audit | Database Audit |
| 14 | Audit Login | SUCCESSFUL\_LOGIN\_GROUP | Y |  |
| 15 | Audit Logout | LOGOUT\_GROUP | Y |  |
| 18 | Audit Server Start & Stop | SERVER\_STATE\_CHANGE\_GROUP | Y |  |
| 20 | Audit Login failed | FAILED\_LOGIN\_GROUP | Y |  |
| 102 | Audit Database Scope GDR | DATABASE\_PERMISSION\_CHANGE\_GROUP | Y | Y |
| 103 | Audit Object GDR | SCHEMA\_OBJECT\_PERMISSION\_CHANGE\_GROUP | Y |  |
| 104 | Audit Add login | SERVER\_PRINCIPAL\_CHANGE\_GROUP | Y |  |
| 105 | Audit login GDR | SERVER\_PERMISSION\_CHANGE\_GROUP | Y |  |
| 106 | Audit login change | SERVER\_PRINCIPAL\_CHANGE\_GROUP | Y |  |
| 107 | Audit login change password | LOGIN\_CHANGE\_PASSWORD\_GROUP | Y |  |
| 108 | Audit Add Login to Server Role | SERVER\_ROLE\_MEMBER\_CHANGE\_GROUP | Y |  |
| 109 | Audit Add DB User | DATABASE\_PRINCIPAL\_CHANGE\_GROUP | Y | Y |
| 110 | Audit Add Member to DB Role Event | DATABASE\_ROLE\_MEMBER\_CHANGE\_GROUP | Y | Y |
| 111 | Audit Add Role | DATABASE\_PRINCIPAL\_CHANGE\_GROUP | Y | Y |
| 112 | Audit App Role Change Event | APPLICATION\_ROLE\_CHANGE\_PASSWORD\_GROUP | Y |  |
| 113 | Audit Statement Permission Event | SCHEMA\_OBJECT\_CHANGE\_GROUP | Y | Y |
| 115 | Audit backup/Restore Event | BACKUP\_RESTORE\_GROUP | Y |  |
| 116 | Audit DBCC Event | DBCC\_GROUP | Y |  |
| 117 | Audit Change Audit Event | AUDIT\_CHANGE\_GROUP | Y |  |
| 118 | Audit Object Derived Permission Event | SCHEMA\_OBJECT\_CHANGE\_GROUP | Y | Y |
| 128 | Audit Database Management Event | DATABASE\_CHANGE\_GROUP | Y | Y |
| 129 | Audit Database Object Management Event | DATABASE\_OBJECT\_CHANGE\_GROUP | Y | Y |
| 130 | Audit Database Principal Management Event | DATABASE\_PRINCIPAL\_CHANGE\_GROUP | Y | Y |
| 131 | Audit Schema Object Management Event | SCHEMA\_OBJECT\_CHANGE\_GROUP | Y | Y |
| 132 | Audit Server Principal Impersonation Event | SERVER\_PRINCIPAL\_IMPERSONATION\_GROUP | Y |  |
| 133 | Audit Database Principle Impersonation Event | DATABASE\_PRINCIPAL\_IMPERSONATION\_GROUP | Y | Y |
| 134 | Audit Server object take ownership event | SERVER\_OBJECT\_OWNERSHIP\_CHANGE\_GROUP | Y |  |
| 135 | Audit Database object take ownership event | DATABASE\_OBJECT\_OWNERSHIP\_CHANGE\_GROUP | Y | Y |
| 152 | Audit Change Database Owner | DATABASE\_OWNERSHIP\_CHANGE\_GROUP | Y | Y |
| 153 | Audit Schema Object Take Ownership Event | SCHEMA\_OBJECT\_OWNERSHIP\_CHANGE\_GROUP | Y | Y |
| 170 | Audit Server Scope GDR | SERVER\_PERMISSION\_CHANGE\_GROUP | Y |  |
| 171 | Audit Server Object Grant Deny Revoke | SERVER\_OBJECT\_PERMISSION\_CHANGE\_GROUP | Y |  |
| 172 | Audit Database Object Grant Deny Revoke | DATABASE\_OBJECT\_PERMISSION\_CHANGE\_GROUP | Y | Y |
| 173 | Audit Server Operation Event | SERVER\_OPERATION\_GROUP | Y |  |
| 175 | Audit Server Alter Trace Event | TRACE\_CHANGE\_GROUP | Y |  |
| 176 | Audit Server Object Management Event | SERVER\_OBJECT\_CHANGE\_GROUP |  |  |
| 177 | Audit server principal management event | SERVER\_PRINCIPAL\_CHANGE\_GROUP | Y |  |
| 178 | Audit Database Operation Event | DATABASE\_OPERATION\_GROUP | Y |  |

With the mappings provided above, a database administrator can create an audit, and an audit specification for each instance of SQL Server, and database audit specification for each database.

Note: Late entry. Anne’s solution for SQL2-00-023900 is a bit complex and may add installation steps. Parking in here for now.

I have created a new SQL policy to checksum the dll’s in the Binn directory for each instance.  This requires that FCIV be installed on each instance and a central working directory be set up on the management server.  A new synonym is created to point to a new table which holds all versions of the checksums.  If the checksums change the insert script will insert a new copy of the files and checksums for the instance, retaining the older versions.  This will allow someone to research as to why the dll’s have changed.



Updated: (Anne) I changed the script to pull the directory instead of hardcoding it.  Also please find report attached.

