**Auditing: SQL Server’s Built-in Traces**

**From:** [**https://msdn.microsoft.com/en-us/library/cc293615.aspx**](https://msdn.microsoft.com/en-us/library/cc293615.aspx)

Aside from the user-defined traces we have discussed throughout this chapter, SQL Server uses tracing to help you audit user activity using a few built-in configurations. In this section, we will briefly discuss the default trace, blackbox traces, and Common Criteria traces—all part of the same SQL Trace infrastructure, but configured and used slightly differently than the traces discussed in the rest of the chapter.

**Default Trace**

Upon first being installed, SQL Server 2005 starts up a background trace known as the *default trace*. If left in place, this trace will have a trace ID of 1, and you can see it in the *sys.traces* view using the following query:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_8c065392-cbed-4da0-a9c5-3c3c5d08ba14');" \o "Copy to clipboard.)

SELECT \*

FROM sys.traces

WHERE id = 1

The trace is fairly lightweight, and includes events for monitoring server starts and stops, object creation and deletion, log and data file auto growth, and other database changes. The information collected by the trace can be retrieved using *fn\_trace\_gettable*, just like any other trace. In addition, data from the default trace is used for various reports that ship with SQL Server Management Studio. The trace is also useful as a general log in case something unexpected happens (“hey, who dropped my table!?”), and for all of these reasons we recommend leaving it in place.

Should you feel the need to disable the default trace, you can do so by changing the *default trace enabled* server option, using the following Transact-SQL:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_0f80b1b7-8228-46fb-a23c-85611372c399');)

EXEC sp\_configure 'default trace enabled', 0

RECONFIGURE

Note that this will immediately stop the trace; there is no need to restart the SQL Server service.

**Blackbox Traces**

Another background trace that SQL Server offers right out of the box is the so-called *blackbox* trace. This trace is designed to behave similarly to an airplane black box, to help you diagnose intermittent server crashes. It is quite a bit heavier than the default trace, and includes the *SP:Starting*, *SQL:BatchStarting*, *Exception*, and *Attention* events, with several columns from each event included in the trace.

This trace is configured by setting the *@options* parameter of *sp\_trace\_create* to a value of 8. The following Transact-SQL can be used to start a blackbox trace:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_730f9f88-a47a-42ea-b3a5-aaea151e4554');)

DECLARE @TraceId INT

EXEC sp\_trace\_create

@TraceId OUTPUT,

@options = 8

EXEC sp\_trace\_setstatus @TraceId, 1

The trace is automatically configured to use two rollover files, and flip back and forth between them when reaching its default maximum file size of 5 MB. However, some customers have reported that 5 MB was not enough back data to help them debug crash problems. To expand the size, you’ll have to specify the *@maxfilesize* parameter, in addition to passing in a value for the *@tracefile* parameter (even a *NULL*). The following Transact-SQL code creates a blackbox trace with a 25 MB maximum size:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_73679940-7b55-48f2-b868-541efda7e20a');)

DECLARE @TraceId int

DECLARE @maxfilesize bigint

SET @maxfilesize = 25

EXEC sp\_trace\_create

@TraceId OUTPUT,

@options = 8,

@tracefile = NULL,

@maxfilesize = @maxfilesize

EXEC sp\_trace\_setstatus @TraceId, 1

By default, the path to the blackbox trace file is in the default SQL Server data file folder. However, just like the file size, the path can also be overridden if necessary, by using the *@tracefile* parameter.

In order to fully take advantage of the blackbox trace if you are facing intermittent problems, you want to make sure that it is always running when your server is running—including after either a planned or unplanned restart. To accomplish this, you can set the blackbox trace to start automatically when SQL Server starts. First, wrap the trace definition in a stored procedure in the *master* database:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_bc51c616-a0a0-4e03-927b-5671701c153c');)

USE master

GO

CREATE PROCEDURE StartBlackBoxTrace

AS

BEGIN

DECLARE @TraceId int

DECLARE @maxfilesize bigint

SET @maxfilesize = 25

EXEC sp\_trace\_create

@TraceId OUTPUT,

@options = 8,

@tracefile = NULL,

@maxfilesize = @maxfilesize

EXEC sp\_trace\_setstatus @TraceId, 1

END

GO

Next, set the procedure to start automatically when the SQL Server service is started:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_b501421f-703c-40cc-bd9b-8622fe0455bf');)

EXEC sp\_procoption

'StartBlackBoxTrace', 'STARTUP', 'ON'

In the event of a crash, you can analyze the collected data to determine what stored procedures or queries were running at the point of failure and—hopefully—figure out what went wrong.

**C2 and Common Criteria Auditing**

SQL Server 2000 added a feature called C2 Audit, which allowed DBAs to meet U.S. government standards for auditing both unauthorized use of and damage to resources and data. A newer, international standard, called Common Criteria, is available in SQL Server 2005 (along with C2, which is still available).

Both of these audit modes utilize SQL Trace to capture a variety of data, including any logins, logouts, access to tables, and many other data points. Both also make use of the value of 4 passed to the *@options* parameter of *sp\_trace\_create*, which automatically shuts down the SQL Server service should any error occur writing to the trace file. This is intended to ensure that even in the case of disk error, none of the audit data will possibly be lost (with the server shut down, no audit data will be generated to lose).

Both C2 and Common Criteria audit modes are enabled or disabled using *sp\_configure*. To enable C2 auditing, configure the C2 Audit Mode option to a value of 1. For Common Criteria, configure the Common Criteria Compliance Enabled option to a value of 1. In both cases, a service restart is required for the auditing to actually start.

[< Back](https://msdn.microsoft.com/en-us/library/cc293614.aspx)      [Next >](https://msdn.microsoft.com/en-us/library/cc293617.aspx)

© Microsoft. All Rights Reserved.