|  |  |  |
| --- | --- | --- |
| **Аудит в SQL Server. Триггер на логон**   |  |  | | --- | --- | | **Рейтинг** |  |   Следующая серия постов основывается на докладе, который я делал на Russian SQL Server User Group в Москве в сентябре 2007 г. Кстати, недавно (в мае) схожую тему затрагивал Дмитрий Костылев (<http://blogs.gotdotnet.ru/personal/DeColores/PermaLink.aspx?guid=400a42fa-9333-476a-b2d5-e9a810954a0b>). Я не могу сослаться на оригинальную сентябрьскую встречу на http://sql.ineta.ru, потому что, похоже, Вадим Осовитный затеял там очередную реконструкцию и пункт "Встречи" из меню исчез. Зато появился блог Вадима. Без сомнения, мне очень интересны его мысли по поводу Silverlight, погоды в Лондоне и жизни вообще. Единственно, непонятно, какое отношение это имеет к Russian SQL Server User Group.  Ладно. Не везет мне с сайтом – повезет где-нибудь еще. Данный пост будет полезен консерваторам, сидящим на SQL Server 2005, а также пользователям SQL Server 2008 не Enterprise (и не Developer) Edition. Короче, где оператора CREATE AUDIT еще или уже не было, а аудит тем не менее провести очень хочется. Вообще, аудит на латыни означает слушание и применяется в мировой практике для обозначения проверки, ревизии. Ровно в том же смысле будем его понимать и мы, только наша бухгалтерия завязана на SQL Server. Т.е. кто зашел на SQL Server, когда, откуда, с какой целью (что делал, чем интересовался, до каких данных хотел долезть и насколько у него это получилось) и т.д. Традиционно для этих целей используются SQL Trace (Profiler), триггеры на DDL / DML, а также читалки лога (покупные или самописные).  Краткая напоминалка по DDL-триггерам. В SQL Server бывают только after. Задаются на двух уровнях: сервер или БД. На каждом уровне фиксирован список событий. События можно посмотреть здесь - <http://msdn.microsoft.com/ru-ru/library/bb522542.aspx>. Всего имеется 13 штук серверных событий  http://i.technet.microsoft.com/hh181042.1(ru-ru,MSDN.10).gif  рис. 1  и 77 штук событий уровня БД:  http://i.technet.microsoft.com/hh181042.2(ru-ru,MSDN.10).gif  рис. 2  Пример DDL-триггера:   1. **use** tempdb   5. **if** **exists**(**select** 1 **from** sys.**triggers** **where** parent\_class\_desc = 'DATABASE' **and** **name** = 'tr1') **drop** **trigger** tr1 **on** **database** 7. go 9. **create** **trigger** tr1 **on** **database** **for** CREATE\_TABLE, ALTER\_TABLE, DROP\_TABLE **as** 11. **select** EventData() 13. raiserror('Все операции над таблицами временно прекращены. Администрация.', 16, 1) 15. **rollback** 17. go   Скрипт 1  Пробуем:   1. **create** **table** t(id **int**)   и получаем ошибку:  http://i.technet.microsoft.com/hh181042.3(ru-ru,MSDN.10).png  рис. 3  Смотрим список таблиц:   1. **select** \* **from** sys.**tables** **where** **type** = 'U'   и видим, что таблица действительно не была создана, т.е. транзакция откатилась.  DDL-триггеры можно создавать не только на одно или несколько событий, но и на группы. А также на отдельные события и группы, на несколько групп и т.д. Группы – это предопределенные объединения родственных событий. См. <http://msdn.microsoft.com/ru-ru/library/bb510452.aspx>.  http://i.technet.microsoft.com/hh181042.4(ru-ru,MSDN.10).gif  рис. 4  Вместо таблиц inserted / deleted информация об окружении в момент срабатывания триггера передается в ф-ции EventData(), которая возвращает XML вида:   1. <EVENT\_INSTANCE> 3. <EventType>CREATE\_TABLE</EventType> 5. <PostTime>2007-09-14T18:59:28.357</PostTime> 7. <SPID>52</SPID> 9. <ServerName> VISTA</ServerName> 11. <LoginName> VISTA\Leshik</LoginName> 13. <UserName>dbo</UserName> 15. <DatabaseName>tempdb</DatabaseName> 17. <SchemaName>dbo</SchemaName> 19. <ObjectName>t</ObjectName> 21. <ObjectType>**TABLE**</ObjectType> 23. <TSQLCommand> 25. <SetOptions ANSI\_NULLS="ON" ANSI\_NULL\_DEFAULT="ON" ANSI\_PADDING="ON" QUOTED\_IDENTIFIER="ON" ENCRYPTED="FALSE" /> 27. <CommandText>**create** **table** t(id **int**)</CommandText> 29. </TSQLCommand> 31. </EVENT\_INSTANCE>   Скрипт 2  Таким образом, при помощи DDL-триггеров и функции EventData() можно получать информацию обо всех перечисленных на рис.1, 2 действиях, а также о том, кто, когда и как их совершил. Остается вопрос с аудитом логона, потому что формально логон не является DDL-операцией - в T-SQL нет команды типа CREATE SESSION. Поэтому в 2005 SP2 был добавлен якобы еще один тип триггера (LOGON), который в BOL в CREATE TRIGGER прописан отдельно от DML- и DDL-триггеров -<http://msdn.microsoft.com/ru-ru/library/ms189799.aspx>. Если не заостряться на формальностях, это обычный DDL-триггер уровня сервера. Когда он заканчивается commitом, SQL Server разрешает соединение, когда rollbackом – отсылает. Смотрим:   1. **use** tempdb  4. **if** **exists**(**select** 1 **from** sys.server\_triggers **where** **name** = 'tr1') **drop** **trigger** tr1 **on** **all** server 6. go 8. **create** **trigger** tr1 **on** **all** server **for** logon **as** 10. **if** original\_login() = 'sa' **commit** -- sa пускаем всегда 12. **declare** **@**MaxConnects **int** 14. **set** **@**MaxConnects = 1 16. **if** (**select** count(1) **from** sys.dm\_exec\_sessions **where** is\_user\_process = 1 **and** original\_login\_name = original\_login()) <= **@**MaxConnects 18. **commit** 20. **else** 22. **rollback** 24. go   Скрипт 3  Триггер на логон срабатывает после фазы аутентификации, но до установления пользовательской сессии, поэтому писать к rollback разные raiserror или print с сообщением, почему он не прошел, говорить ай-яй-яй, низя, показывать язык и пр. бессмысленно – юзер все равно ничего этого не увидит.  Вот обещаный аудит логона:   1. **if** **exists**(**select** 1 **from** sys.**tables** **where** **name** = ' ЖурналЛогонов') **drop** **table** ЖурналЛогонов 3. **create** **table** ЖурналЛогонов 5. (id **int** identity, когда **datetime**, куда sysname, кто sysname, тип\_события sysname, спид **int**, клиентская\_тачка sysname **null**, прикладуха sysname **null**)   9. **if** **exists**(**select** 1 **from** sys.server\_triggers **where** **name** = 'tr1') **drop** **trigger** tr1 **on** **all** server 11. go 13. **create** **trigger** tr1 **on** **all** server **for** logon **as** 15. **declare** **@**eventdata xml 17. **set** **@**eventdata = EventData() 19. **insert** tempdb.dbo. ЖурналЛогонов (когда, куда, кто, тип\_события, спид, клиентская\_тачка, прикладуха) 21. **values** ( 23. **@**eventdata.value('(/EVENT\_INSTANCE/PostTime)[1]', 'datetime'), 25. **@**eventdata.value('(/EVENT\_INSTANCE/ServerName)[1]', 'sysname'), 27. **@**eventdata.value('(/EVENT\_INSTANCE/LoginName)[1]', 'sysname'), 29. **@**eventdata.value('(/EVENT\_INSTANCE/EventType)[1]', 'sysname'), 31. **@**eventdata.value('(/EVENT\_INSTANCE/SPID)[1]', 'int'), 33. **@**eventdata.value('(/EVENT\_INSTANCE/ClientHost)[1]', 'sysname'), 35. App\_Name() 37. ) 39. **commit**   Скрипт 4  Выполняем скрипт 4 и тестируем созданный триггер, организовав новую сессию:  http://i.technet.microsoft.com/hh181042.5(ru-ru,MSDN.10).png  рис. 5  http://i.technet.microsoft.com/hh181042.6(ru-ru,MSDN.10).png  рис. 6  Видим, что попытки логона теперь фиксируются в указанной таблице:  http://i.technet.microsoft.com/hh181042.7(ru-ru,MSDN.10).png  рис. 7  Вопрос: как модифицировать предыдущий скрипт, если мы хотим клиента не пущать, а его умысел залогиниться всеж-таки зафиксировать? Элементарно: надо вместо commit поставить rollback:   1. **truncate** **table** tempdb.dbo.ЖурналЛогонов  4. **if** **exists**(**select** 1 **from** sys.server\_triggers **where** **name** = 'tr1') **drop** **trigger** tr1 **on** **all** server 6. go 8. **create** **trigger** tr1 **on** **all** server **for** logon **as** 10. **declare** **@**eventdata xml 12. **set** **@**eventdata = EventData() 14. **rollback** --Откатываем = не пущаем 16. **begin** tran --транзакцию вставки в журнал фиксируем 18. **insert** tempdb.dbo.ЖурналЛогонов (когда, куда, кто, тип\_события, спид, клиентская\_тачка, прикладуха) 20. **values** ( 22. **@**eventdata.value('(/EVENT\_INSTANCE/PostTime)[1]', 'datetime'), 24. **@**eventdata.value('(/EVENT\_INSTANCE/ServerName)[1]', 'sysname'), 26. **@**eventdata.value('(/EVENT\_INSTANCE/LoginName)[1]', 'sysname'), 28. **@**eventdata.value('(/EVENT\_INSTANCE/EventType)[1]', 'sysname'), 30. **@**eventdata.value('(/EVENT\_INSTANCE/SPID)[1]', 'int'), 32. **@**eventdata.value('(/EVENT\_INSTANCE/ClientHost)[1]', 'sysname'), 34. App\_Name() 36. ) 38. **commit**   Скрипт 5  При попытке логона клиент получит отлуп:  http://i.technet.microsoft.com/hh181042.8(ru-ru,MSDN.10).png  рис. 8  Меж тем его попытка будет отражена в таблице:  http://i.technet.microsoft.com/hh181042.9(ru-ru,MSDN.10).png  рис. 9  Вопрос: что будет, если я сейчас выйду из текущей сессии, а триггер останется висеть и посылать всех на? Или, скажем, я допустил ошибку, когда его писал. Имеется в виду не синтаксическая, а времени выполнения. Например, укажем в INSERT несуществующую таблицу. При возникновении run-time error выполнение триггера прекращается и делается неявный откат вместо ошибочного оператора. То есть вместо INSERT получим ошибку и ROLLBACK. Это тоже означает все в сад. Предположим, посмотреть текст ошибки можно SQLном ErrorLoge. Кто забыл, где находится директория с журналами, может обратиться к<http://blogs.msdn.com/alexejs/archive/2009/06/25/9803218.aspx>.  http://i.technet.microsoft.com/hh181042.10(ru-ru,MSDN.10).png  рис. 10  Теперь, когда ошибка понятна, возникает вопрос, как ее исправить, если мы уже вышли из редактора, потому что триггер никому не даст залогиниться на сервер.  Два возможных пути для этого имеются. Первый путь - перестартовать SQL Server в минимальной конфигурации. Например, из одного командного окна делаем:  net stop mssqlserver  “C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Binn\sqlservr.exe” –c –f  А из другого:  sqlcmd  drop trigger tr1 on all server.  Если забыли, как он, треклятый, называется, воспользуйтесь select \* from sys.server\_triggers.  Второй путь – это DAC (выделенное административное соединение) - <http://msdn.microsoft.com/ru-ru/library/ms178068.aspx>. SSMS сначала будет пытаться приконнектить Object Explorer (если он был открыт), который не поддерживает DAC. Забейте на него, и соединяйте окно запросов. Из окна запросов нужно прикончить глюкавый триггер, после чего на сервер станет возможно логиниться обычным порядком. |