Extended Events or SQL Profiler

Last updated by | Vitor Tomaz | Nov 22, 2021 at 11:51 AM PST

Contents

Overview

Extended events (XEvents) provide a way for you to trace what happens to your server. One of the functionalities is that it can track every query that goes into your azure SQL database. In this document, we will document steps to filter queries that consume CPU over certain threshold for troubleshooting.

Caution when using Extended events

For SQL DB, extended events require you to use remote azure storage because end user doesn't have direct access to local storage. Volume of data is directly related to how busy your server is and filters you apply to the xevents. If your application has high volume of queries, it's best to do filtering to specifically what you need. High volume (busy) database generates many events, large volume of unfiltered tracing could cause delay writing to remote storage and cause query to slowdown.

While doing tracing, monitor your application behavior very closely and stop immediately.

Keep tracing short if possible and stop it as soon as you get what you need

High level steps

a. Storage account: Create an azure storage account and a blob container and Generate a SAS token from the storage account. This SAS token is needed for creating database credential.
b. Create a database credential
c. Create Xevent sessions
d. Let your app run
e. Stop xevent session
f. Download the file
g. Analyze using SQL Server management studio SSMS 17.3

Create storage account & obtain SAS token

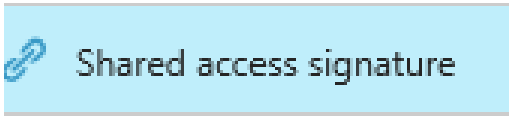
1. Create storage account and container

Create an resource manager azure storage account in the same region as your SQL Database and BLOB container called XEvent

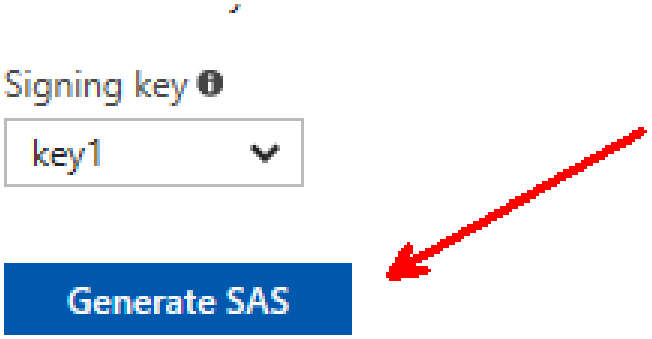
See https://docs.microsoft.com/en-us/azure/storage/common/storage-create-storage-account

2. Obtain shared access signature (SAS token)

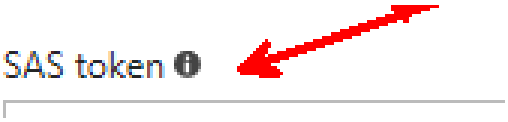
a. Got to the storage account you just created
b. Click on "Shared Access Signature"



c. Choose start and expiry date/time
d. Click on generate SAS



e. Copy the SAS token for later usage



Note that when you copy the SAS token, it starts with question mark ?. You do NOT need this question mark (?). If you keep it, the credential you create next step will not work.

3. Create database credential

--create master key

CREATE MASTER KEY ENCRYPTION BY PASSWORD='<EnterStrongPasswordHere>';

-- when you copy SAS, there is a ? in the beginnging remove that

CREATE database scoped CREDENTIAL [https://<storageacountname>.blob.core.windows.net/xevent]

WITH IDENTITY='SHARED ACCESS SIGNATURE'

, SECRET = 'sv=.....' --make sure you remove ? In the SAS token copied from previous step

4. Create Xevent event session

1. Download management studio

Instructions here requires SQL Server management studio 17.3. download and install SSMS 17.3 from

https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms

2. Pick a threshold for cpu_time

The event script below has 15000000 in two places for cpu_time. CPU_time is in microseconds. 15000000, means you want to track any queries that consume more than 15 seconds of cpu.

Choose a threshold that works for you and convert it to microseconds.

3. Modify and create xevent
    1. Modify the script

You need to make the following change:

filename=N'https://<yourstorageaccount>.blob.core.windows.net/xevent/batch_rpc.xel'). you need to change the filename to match to the storage account you created in the beginning

```
CREATE EVENT SESSION [batch_rpc] ON DATABASE

ADD EVENT sqlserver.error_reported (     ACTION (package0.event_sequence, sqlserver.client_app_name, sqlserver.client_hostname, sqlserver.client_pid, sqlserver.database_id, sqlserver.database_name, sqlserver.query_hash, sqlserver.request_id,   sqlserver.session_id,  sqlserver.sql_text, sqlserver.transaction_id, sqlserver.username)     ),

ADD EVENT sqlserver.sql_batch_completed ( SET collect_batch_text=(1)      ACTION (package0.event_sequence, sqlserver.client_app_name, sqlserver.client_hostname, sqlserver.client_pid, sqlserver.database_id, sqlserver.database_name,  sqlserver.query_hash, sqlserver.request_id,  sqlserver.session_id,  sqlserver.sql_text, sqlserver.transaction_id, sqlserver.username) ),

ADD EVENT sqlserver.rpc_completed ( SET collect_data_stream=(1)      ACTION (package0.event_sequence, sqlserver.client_app_name, sqlserver.client_hostname, sqlserver.client_pid, sqlserver.database_id, sqlserver.database_name,   sqlserver.query_hash, sqlserver.request_id,   sqlserver.session_id,  sqlserver.sql_text, sqlserver.transaction_id, sqlserver.username)  ),

ADD EVENT sqlserver.error_reported(    ACTION(sqlserver.session_id,sqlserver.sql_text))

ADD TARGET package0.event_file(SET filename=N'https://<yourstorageaccount>.blob.core.windows.net/xevent/batch_rpc.xel')

WITH (MAX_MEMORY=4096 KB,EVENT_RETENTION_MODE=ALLOW_SINGLE_EVENT_LOSS,MAX_DISPATCH_LATENCY=10 SECONDS,MAX_EVENT_SIZE=0 KB,MEMORY_PARTITION_MODE=PER_CPU,TRACK_CAUSALITY=OFF,STARTUP_STATE=OFF)

Go
```

2. Run the script to create xevent session

Use SSMS 17.3 and connect to your SQL DB. Make sure you connect to the user database (not master). Run the script (after modification) above to create xevent session

3. Start xevent

alter event session [batch_rpc] ON DATABASE state = start

5. Wait for the app to run

wait for app to run for a while to duplicate your issue

stop event session: alter event session [batch_rpc] ON DATABASE state = stop

6. Download the file

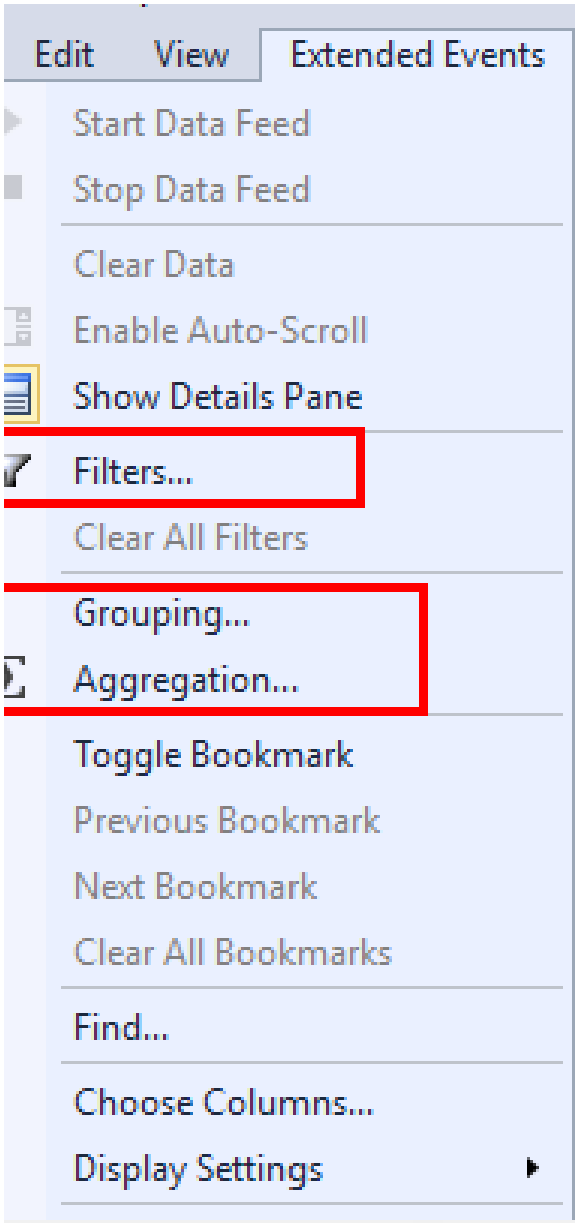
you can go to the storage account and the container to download the file

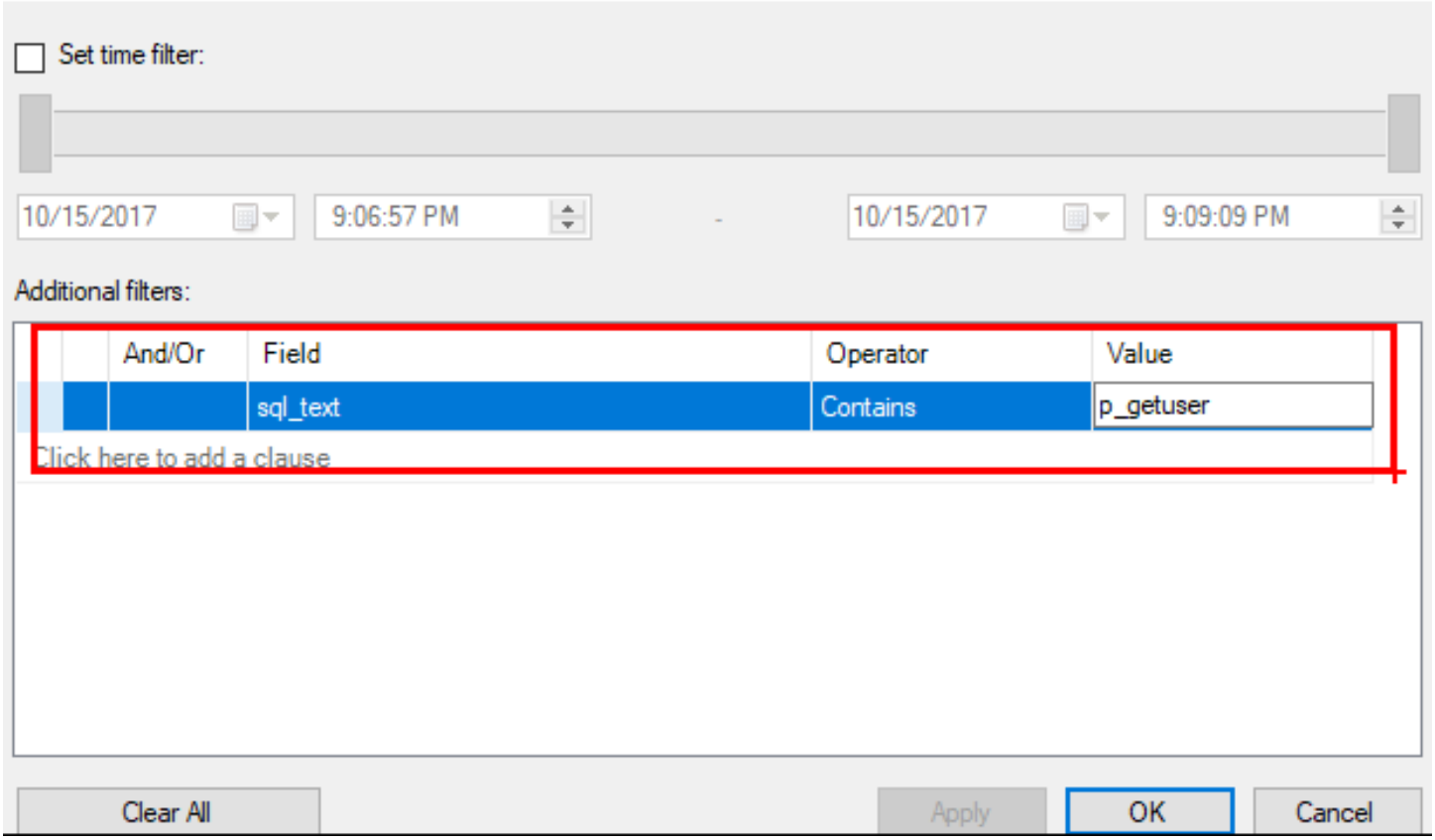
if you click on the file, top corner has a download button

7. Use SSMS to analyze the file

analyze the xevent file using SSMS 17.3

Load the file using SSMS and click on "Extended Events" menu. Choose "filters" and filter on the query text of your interest


Edit
View
Extended Events
Start Data Feed
Stop Data Feed
Clear Data
Enable Auto-Scroll
Show Details Pane
Filters...
Clear All Filters
Grouping...
Aggregation...
Toggle Bookmark
Previous Bookmark
Next Bookmark
Clear All Bookmarks
Find...
Choose Columns...
Display Settings



You can see CPU usage in microseconds and full SQL text that has parameters used

Details

Field	Value
batch_text	exec p_getuser N'%xyz%'
client_app_name	Microsoft SQL Server Management Studio - Query
client_hostname	JACKLIMOBILE
client_pid	4456
cpu_time	57453000
database_id	5
database_name	jacklidb
duration	126754603
event_sequence	10
logical_reads	277489
physical_reads	0
query_hash	0
request_id	0
result	OK
row_count	18
session_id	125
spills	0
sql_text	exec p_getuser N'%xyz%'
transaction_id	0
username	azureadmin
writes	0

How good have you found this content?